

## Задача А. Разреженное дерево Мёркла. Проверка.

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вам нужно реализовать построение разреженного дерева Мёркла и проверить корректность доказательств для блоков.

Дерево Мёркла описывается тремя алгоритмами (производными от алгоритмов, рассмотренных в лекции):

- $(d, T) \leftarrow \text{setup}(X)$  — строит дерево  $T$  и его короткое подтверждение (хеш корневого узла)  $d$  по блокам данных  $X = \{x_0, x_1, \dots, x_{n-1}\}$
- $(\pi_i, x_i) \leftarrow \text{prove}(T, X, i)$  — строит доказательство для дерева  $T$  по индексу блока данных  $i$ . Результатом является пара из блока данных  $x_i$  и хешей соседей на пути от соответствующего листа дерева до корня дерева (не включая корня).
- $\text{valid} \leftarrow \text{verify}(d, i, \pi_i, x_i)$  — проверяет, является ли блок данных частью дерева, по соответствующему подтверждению  $d$ , блоку  $x_i$ , его индексу  $i$ , и соседям по пути от листа до корня дерева  $\pi_i$ .

Дерево снизу вверх строится следующим образом: если блока данных с определенным индексом на входе нет, то лист дерева равен `null`, иначе лист вычисляется как `SHA256(0||xi)`, где 0 это нулевой байт (8 нулевых бит). Для внутренних узлов хеш вычисляется как `SHA256(1||dig(l)||2||dig(r))`, где `dig(l)` есть хеш левого потомка, `dig(r)` — хеш правого потомка (числа 1 и 2 тоже занимают 8 бит). Если оба потомка равны `null`, то внутренний узел тоже равен `null`. Если же ровно один из потомков из потомков равен `null`, то соответствующий `dig(·)` заменяется пустой строкой.

В этой задаче требуется проверить доказательства для блоков на корректность.

### Формат входных данных

В первой строке файла задана высота дерева  $h$  ( $1 \leq h \leq 30$ ). Вторая строка содержит хеш корня дерева в `Base64`. Третья строка содержит  $q$  ( $1 \leq q \leq 10\,000$ ) — число блоков, для которых мы хотим проверить доказательства. Затем следует  $q$  доказательств. Каждое доказательство в первой строке содержит идентификатор блока ( $0 \leq id \leq 2^h - 1$ ) и данные блока в `Base64`. Следующие  $h$  строк содержат хеши соседей в `Base64` снизу вверх.

### Формат выходных данных

Для каждого доказательства выведите в одной строке “YES”, если оно корректно, и “NO” в противном случае.

## Пример

стандартный ввод
3 mjlPuB+DhKr1xL1MLyG/OM7iog2GgjquyVsmPxyD0G8= 4 0 null wrWzhwQpSS/S3/9DoKYhX9X2ESfcEAd24GvbAtTVaw4= rBdCkLCyPdT17nxs9ubPhQyNlgrokFTD9xdYUNPnjSE= null 1 acab null rBdCkLCyPdT17nxs9ubPhQyNlgrokFTD9xdYUNPnjSE= null 2 cacx null c6WRavDoPHscMlVKF9xYcFZBz1b83qinYC1AqQcE+i0= null 3 null hpnxjn69vVsDnfP/00851ndk5gU1yD9exgwh/+oyjL8= c6WRavDoPHscMlVKF9xYcFZBz1b83qinYC1AqQcE+i0= null
стандартный вывод
NO YES NO YES

## Задача В. Разреженное дерево Мёркла. Доказательства.

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мегабайт

Вам нужно реализовать построение разреженного дерева Мёркла и проверить корректность доказательств для блоков.

Дерево Мёркла описывается тремя алгоритмами (производными от алгоритмов, рассмотренных в лекции):

- $(d, T) \leftarrow \text{setup}(X)$  — строит дерево  $T$  и его короткое подтверждение (хеш корневого узла)  $d$  по блокам данных  $X = \{x_0, x_1, \dots, x_{n-1}\}$
- $(\pi_i, x_i) \leftarrow \text{prove}(T, X, i)$  — строит доказательство для дерева  $T$  по индексу блока данных  $i$ . Результатом является пара из блока данных  $x_i$  и хешей соседей на пути от соответствующего листа дерева до корня дерева (не включая корня).
- $\text{valid} \leftarrow \text{verify}(d, i, \pi_i, x_i)$  — проверяет, является ли блок данных частью дерева, по соответствующему подтверждению  $d$ , блоку  $x_i$ , его индексу  $i$ , и соседям по пути от листа до корня дерева  $\pi_i$ .

Дерево снизу вверх строится следующим образом: если блока данных с определенным индексом на входе нет, то лист дерева равен `null`, иначе лист вычисляется как `SHA256(0||xi)`, где 0 это нулевой байт (8 нулевых бит). Для внутренних узлов хеш вычисляется как `SHA256(1||dig(l)||2||dig(r))`, где `dig(l)` есть хеш левого потомка, `dig(r)` — хеш правого потомка (числа 1 и 2 тоже занимают 8 бит), если строго один из потомков из потомков равен `null`, то `dig(·)` заменяется пустой строкой, если оба потомка равны `null`, то внутренний узел тоже есть `null`.

В этой задаче нужно представить доказательства для запрошенных блоков.

### Формат входных данных

В первой строке файла задана высота дерева  $h$  ( $1 \leq h \leq 30$ ). Вторая строка содержит число блоков  $n$  ( $1 \leq n \leq 1000$ ). Далее следует описания блоков по одному на строке: идентификатор блока ( $0 \leq id \leq 2^h - 1$ ) и данные блока, записанные в `Base64`. Все идентификаторы различны. Данные блока не равны `null`.

Затем следует число блоков  $q$  ( $1 \leq q \leq 1000$ ), для которых мы хотим узнать доказательство. На следующей строке через пробел заданы идентификаторы блоков (различные числа от 0 до  $2^h - 1$ ).

### Формат выходных данных

Для каждого блока выведите его доказательство. Первая строка должна содержать идентификатор блока и данные блока в `Base64`. Затем должно следовать  $h$  строк — `Base64` запись хешей соответствующих соседей, начиная снизу.

## Пример

стандартный ввод
3 2 1 acab 2 caca 4 0 1 2 3
стандартный вывод
0 null wrWzhwQpSS/S3/9DoKYhX9X2ESfcEAd24GvbAtTVtw4= rBdCkLCyPdT17nxs9ubPhQyNlgrokFTD9xdYUNPnjSE= null 1 acab null rBdCkLCyPdT17nxs9ubPhQyNlgrokFTD9xdYUNPnjSE= null 2 caca null c6WRavDoPHscMlVKF9xYcFZBz1b83qinYC1AqQcE+i0= null 3 null hpnxjn69vVsDnfP/00851ndk5gU1yD9exgwh/+oyjL8= c6WRavDoPHscMlVKF9xYcFZBz1b83qinYC1AqQcE+i0= null