Задание 7. Структуры данных и разные задачи

1 Назовем первый стек `vault`, второй `out`. Стек `vault` будет использоваться для хранения элементов, `out` — для их вывода. Метод `push(a)` будет просто записывать элемент `a` в стек `vault`. При вызове метода `pop` может произойти два случая:

(1) Стек `out` не пуст. Тогда применяем к нему `pop` как для стека.

(2) Стек `out` пуст. Тогда к каждому элементу `vault` последовательно применяем `pop` стека и записываем каждый элемент в стек `out`. Заметим, что после этой операции элементы `out` стоят в обратном порядке, относительно их позиции в `vault`. Таким образом, если последовательно применять к ним метод `pop` стека `out`, элементы будут выходить в порядке попадания в `vault`, т.е. как в очереди. Остается один раз применить `pop`.

Ясно, что метод `push(a)` работает за $O(1)$. Метод `pop` работает в худшем случае за $O(n)$.

2 Положим $y(n)$ равным y после работы алгоритма для $N = 1\ldots1$ с n единицами, т.е. $N = 2^n - 1$. Посчитаем чему равно $y(n+1)$. Оно равно $y(n)$ плюс $k(x)$ для всех x, состоящих из $n+1$ цифры. Заметим, что каждому такому x можно однозначно сопоставить $f(x) < 2^n$, для этого достаточно убрать ведущую единицу и нули, следующие за ней. Обратно, каждому $\widetilde{x} < 2^n$ однозначно сопоставляется $2^n \leq f^{-1}(\widetilde{x}) < 2^{n+1}$ добавлением ведущих нулей и единицы. Теперь заметим, что равенство $k(x) = k\big(f(x)\big)$ выполняется для всех $2^n < x < 2^{n+1}$ так как у них в записи есть единицы помимо ведущей, которые остаются на месте при отображении f. То есть, равенство не выполняется только для $x = 2^n = 10\ldots0$, так как ему соответствует $f(x) = 0$ и $k(x) = n$, $k(0) = 0$(начинаем с 1, поэтому полагаем $k(0) = 0$).

Итак, $y(n+1) = y(n) + n + y(n)$, здесь слагаемые соответствуют $1 \leq x \leq 2^n - 1$, $x = 2^n$, $2^n + 1 \leq x \leq 2^{n+1} - 1$. Добавим начальное условие $y(1) = k(1) = 0$ и получаем ЛНРУ 1-го порядка. Общее решение однородного уравнения $y(n+1) - 2y(n) = 0$ — $y_0(n) = C \cdot 2^n$.

Из вида правой части, надо искать частное решение в виде $y_1 = a_0 + a_1 n$. Подставим:

$$a_0 + a_1 n + a_1 - 2a_0 - 2a_1 n = n.$$

Получили два уравнения на коэффициенты:

$$\begin{cases} a_0 + a_1 - 2a_0 = 0 \\ a_1 - 2a_1 = 1 \end{cases} \Rightarrow \begin{cases} a_0 = -1 \\ a_1 = -1 \end{cases}$$

Итак, частное решение — $y_1(n) = -n - 1$ и общее решение неоднородного равно

$$y(n) = y_0(n) + y_1(n) = C2^n - n - 1.$$

Подставляем $y(1) = 0$ и получаем $2C - 3 = 0$, откуда $C = \frac{3}{2}$. Считая, что $N = 2^n$, получаем $y = y(n) + n = y\left(\log_2 N\right) + \log_2 N = \frac{3}{2}N - 1$

Ответ: $y = \frac{3}{2}N - 1$.

4 Рассмотрим броски первого шарика. Если он не разбился при броске, значит нет необходимости проверять все этажи ниже. Далее считаем, что каждый следующий бросок первого шарика был с более высокого этажа, чем предыдущий.

Положим k_i равным количеству этажей между i-м броском первого шарика и $i+1$-м, дополнительно k_0 равно этажу, первого броска минус 1. Например, если шарик бросали с этажей $3, 7, 19$, то $k_0 = 2, k_1 = 3, k_2 = 11$.

Далее, если шарик разбился на i-м броске, надо проверить k_i этажей вторым шариком, т.к. любой из этих этажей может быть тем, на котором шарик разбивается. Например, если шарик не разбился на 3 этаже, но разбился на 7, надо проверить $4, 5, 6$ в таком порядке. Итак, необходимо $N + \max_{1\leq i\leq N} k_i$. Так как требуется максимум, будем считать все k_i равными k. Тогда получаем тождество

$$(k+1)N = 100.$$

Задача минимизации $N + k$ эквивалентна задаче минимаизации $N + k + 1 = N + \widetilde{k}$. Итак, задача свелась к стандартно задаче

$$\begin{cases} \min N + \widetilde{k}, \\ N\widetilde{k} = 100. \end{cases}$$

Отсюда $N = \widetilde{k} = 10$ и $N + k = 19$.

Алгоритм: Бросаем первый шарик на каждом десятом этаже, когда он разбивается, бросаем второй на каждом этаже между двумя последними бросками. **Оценка на количество бросков:** 19.