High Score

Идея: Дмитрий Якутов
Разработка: Дмитрий Якутов

Посмотрим на состояние игры в произвольный момент времени. Пусть в мультисете в данный момент есть число $a = 2^t$. Сколько очков получил игрок за создание этого числа с нуля? Если $a = 2$, то игрок не получил очков. Иначе рассмотрим все числа, которые игрок добавлял в мультисет с помощью операции Insert, которые внесли вклад в текущее число a:

- Если при операциях Insert игрок добавлял только числа со значением 2, то игрок получил $Q(t) = (t-1) \cdot 2^t$ очков: при операциях Merge с $x = 2$ игрок суммарно получил 2^t очков, с $x = 4$ – тоже 2^t очков, $\ldots$, с $x = a$ – тоже 2^t очков;
- Аналогично, если при операциях Insert игрок добавлял только числа 4, то игрок суммарно получил $P(t) = (t-2) \cdot 2^t$ очков;
- Если часть операций Insert выполнялась со значением 2, а остальные – со значением 4, то игрок мог получить любое значение от $P(t)$ до $Q(t)$ с шагом 4 очка.

Из рассуждений выше есть одно исключение: если $t = k + 1$, то есть речь идет о максимальном возможном значении числа в мультисете. В этом случае невозможно получить это число, добавляя только 2 при Insert, так как для этого в какой-то момент на доске должны быть числа: $2, 2, 4, \ldots 2^k$, то есть $k + 1$ число. Максимальное возможное число очков, полученное при создании 2^{k+1} – это $R(t) = Q(t) - 4 = (t-1) \cdot 2^t - 4$ очка.

Приведем алгоритм, с помощью которого можно брать любое число очков от 0 до $MaxScore(k) = \sum\limits_{t=2}^{k+1} R(t)$, методом математической индукции по значению k.

Если $k = 2$, то максимальный возможный счет $\sum\limits_{t=2}^{3}((t-1) \cdot 2^t - 4) = (1 \cdot 4 - 4) + (2 \cdot 8 - 4) = 0 + 12 = 12$. Способ набрать 12 очков проиллюстрирован в пояснении к сэмплу.

При $k \geq 3$. Пусть мы хотим набрать ровно h очков, где $h \mod 4 = 0$:

- Если $4 \leq h \leq MaxScore(k-1)$, то наберем нужные очки, воспользовавшись индукционным предположением. Кол-во чисел в мультисете в любой момент времени не будет превышать $k - 1$, что нам подходит;
- Если $MaxScore(k-1) + 4 \leq h \leq MaxScore(k-1) + Q(k)$, то сначала сделаем так, в мультисете было только одно число 2^k, набрав максимальный возможный для этого числа счет $Q(k) = (k-1) \cdot 2^k$ при создании этого числа, а затем доберем оставшиеся очки, используя не более $k - 1$ слотов в мультисете. Это можно сделать, так как $MaxScore(k-1) + 4 \geq Q(k)$;
- Если $MaxScore(k-1) + Q(k) + 4 \leq h \leq MaxScore(k-1) + P(k+1)$, то сначала сделаем число 2^{k+1}, набрав счет $P(k+1)$, а затем доберем оставшиеся очки, воспользовавшись индукционным предположением. Это мы можем сделать, так как $MaxScore(k-1) + Q(k) + 4 \geq P(k+1)$.
- Если $MaxScore(k-1) + P(k+1) \leq h \leq MaxScore(k-1) + R(k+1) = MaxScore(k)$, то сначала сделаем число 2^{k+1}, набрав ровно $h_1 = h - MaxScore(k-1)$ очков, а затем доберем оставшиеся $MaxScore(k-1)$ очков. Это мы можем сделать, так как $P(k+1) \leq h_1 \leq R(k+1)$.

Таким образом мы можем набрать любое число очков от 0 до $MaxScore(k)$ с шагом 4 очка.

На практике следующий жадный алгоритм приводит к тому же результату. Пока мы не набрали нужное число очков, будем находить максимальное t такое, что $P(t)$ не превосходит оставшегося числа очков, и создавать 2^t, вычитая максимальное возможное число очков, правильно выбирая между $Q(t)$ и $R(t)$.