Problem Tutorial: "Bruteforce"

Note that we can rewrite $\left\lfloor \frac{b_i \cdot i^k}{w} \right\rfloor$ as $\frac{b_i \cdot i^k - (b_i \cdot i^k) \bmod w}{w}$. So in some sense we have two independent problems: finding $\sum_{i=1}^{n} b_i \cdot i^k$ and $\sum_{i=1}^{n} (b_i \cdot i^k) \bmod w$. We will solve both problems using segment tree. We will build our segment tree over values(so it will have fixed size of 10^5), soeach node will maintain information about some subsegment of values. Let's talk about second subproblem first: for each node we will maintain $cnt[x][y]$ — how many numbers c in this node have $b_c \bmod w = x$ and $c \bmod w = y$. When we compute this value, we consider only numbers stored in this node(so number c ranges from 1 to amount of numbers lying in this node). As usual, the hardest thing that we need to do is to be able to merge information from two sons. In this subproblem, it's easy to do this, because the only thing that happens is change of indexation for right son(each index will be shifted by amount of numbers in left son). So we are able to do single merge in $O(w^2)$ time.

To solve first subproblem, we will maintain $f[t]$ — $\sum_{i=1}^{n} (b_i \cdot i^t)$ for $0 \le t \le k$. Then, to do merge we should be able to compute $\sum_{i=1}^{n} (b_i \cdot (i + shift)^t)$ for some integer value of $shift$. It's easy to see, that after opening brackets, it will be equal to $\sum_{i=1}^{n} \sum_{x=0}^{t} (b_i \cdot i^x \cdot \binom{t}{x} \cdot shift^{t-x}) = \sum_{x=0}^{t} f[x] \cdot \binom{t}{x} \cdot shift^{t-x}$. So, we can do merge in $O(k^2)$.

Each query changes only one leaf node in segment tree, so we can do single update in $O((k^2 + w^2) \cdot log(10^5)$ time. So complexity is $O((q + n) \cdot (k^2 + w^2) \cdot log(10^5))$

One small note — in terms of implementation, it's better to add value $b_0 = 0$ and solve problem in 0-indexation.