NOI 2019 省队组队选拔赛（模拟赛）题解

租酥雨

2019 年 3 月 12 日

得分情况

内容…

吐槽环节

算法 $1,2$

算法 1：直接 $O(\binom{n}{2}^m \times n)$ 暴搜即可。

算法 $1,2$

算法 1：直接 $O(\binom{n}{2}^m \times n)$ 暴搜即可。
算法 2：从后往前倒着搜。由于后染的位置是不会被先染的覆盖掉的，所以状态数会相对较少一些。
可能 2.0 秒搜不完，但是可以打表呀！

算法 $1, 2$

算法 1：直接 $O(\binom{n}{2}^m \times n)$ 暴搜即可。
算法 2：从后往前倒着搜。由于后染的位置是不会被先染的覆盖掉的，所以状态数会相对较少一些。
可能 2.0 秒搜不完，但是可以打表呀！
期望得分 10 至 20 分。

算法 3

不考虑先前染的颜色被覆盖这件事情。如果某种颜色在最终的序列中出现了 x 次，那么我们就直接认为在染这种颜色的时候，我们只染了 x 个格子。

算法 3

不考虑先前染的颜色被覆盖这件事情。如果某种颜色在最终的序列中出现了 x 次，那么我们就直接认为在染这种颜色的时候，我们只染了 x 个格子。

但这样一来每次染色的格子就不再是连续的一段了。不过如果我们把给一段格子染色认为是在已被染色的颜色序列中插入一段，那么一切都显得简单而明晰了！

算法 3

不考虑先前染的颜色被覆盖这件事情。如果某种颜色在最终的序列中出现了 x 次，那么我们就直接认为在染这种颜色的时候，我们只染了 x 个格子。

但这样一来每次染色的格子就不再是连续的一段了。不过如果我们把给一段格子染色认为是在已被染色的颜色序列中插入一段，那么一切都显得简单而明晰了！

直接设 $f_{i,j}$ 表示前 i 次染色，已有颜色序列长度为 j 的方案数，转移只需要枚举这次插入的颜色段长度即可。

$$f_{i,j} = f_{i-1,j} + \sum_{k=0}^{j-1} f_{i-1,k} \times (k+1)$$

注意最后一次染色的长度必须非 0，也就是说 $f_{m,j}$ 不能从 $f_{m-1,j}$ 转移过来。答案就是 $f_{m,n}$。

算法 3

不考虑先前染的颜色被覆盖这件事情。如果某种颜色在最终的序列中出现了 x 次，那么我们就直接认为在染这种颜色的时候，我们只染了 x 个格子。
但这样一来每次染色的格子就不再是连续的一段了。不过如果我们把给一段格子染色认为是在已被染色的颜色序列中插入一段，那么一切都显得简单而明晰了！
直接设 $f_{i,j}$ 表示前 i 次染色，已有颜色序列长度为 j 的方案数，转移只需要枚举这次插入的颜色段长度即可。

$$f_{i,j} = f_{i-1,j} + \sum_{k=0}^{j-1} f_{i-1,k} \times (k+1)$$

注意最后一次染色的长度必须非 0，也就是说 $f_{m,j}$ 不能从 $f_{m-1,j}$ 转移过来。答案就是 $f_{m,n}$。可以使用前缀和优化至 $O(n^2)$。

算法 4

转移中 $f_{i,j}$ 从 $f_{i-1,j}$ 转移来的这部分很烦，能不能去掉它？

算法 4

转移中 $f_{i,j}$ 从 $f_{i-1,j}$ 转移来的这部分很烦，能不能去掉它？
显然是可以的，只要给最终的答案乘上个组合数就行了。

算法 4

转移中 $f_{i,j}$ 从 $f_{i-1,j}$ 转移来的这部分很烦，能不能去掉它？
显然是可以的，只要给最终的答案乘上个组合数就行了。
枚举实际染上了 k 种颜色，可以发现方案数就是 $\prod_{i=2}^{n}(x+i)$ 的第 $n-k$ 次项系数。

算法 4

转移中 $f_{i,j}$ 从 $f_{i-1,j}$ 转移来的这部分很烦，能不能去掉它？
显然是可以的，只要给最终的答案乘上个组合数就行了。
枚举实际染上了 k 种颜色，可以发现方案数就是 $\prod_{i=2}^{n}(x+i)$ 的第 $n-k$ 次项系数。
你可以通过 dp 式直接看出来，也可以根据具体含义理解。

算法 4

转移中 $f_{i,j}$ 从 $f_{i-1,j}$ 转移来的这部分很烦，能不能去掉它？
显然是可以的，只要给最终的答案乘上个组合数就行了。
枚举实际染上了 k 种颜色，可以发现方案数就是 $\prod_{i=2}^{n}(x+i)$ 的第 $n-k$ 次项系数。
你可以通过 dp 式直接看出来，也可以根据具体含义理解。
假设实际染了 k 次颜色，每次染色前已有颜色序列长度为 $a_1, a_2 \ldots a_k$，那么就一定有 $0 = a_1 < a_2 < \ldots < a_k < n$，而这部分对答案的贡献是 $\prod_{i=1}^{k}(a_i+1)$，所以总答案就是从 $[2, n]$ 中选出 $k-1$ 个数乘起来的所有方案的乘积之和。

算法 4

转移中 $f_{i,j}$ 从 $f_{i-1,j}$ 转移来的这部分很烦，能不能去掉它？
显然是可以的，只要给最终的答案乘上个组合数就行了。
枚举实际染上了 k 种颜色，可以发现方案数就是 $\prod_{i=2}^{n}(x+i)$ 的第 $n-k$ 次项系数。
你可以通过 dp 式直接看出来，也可以根据具体含义理解。
假设实际染了 k 次颜色，每次染色前已有颜色序列长度为 $a_1, a_2 \ldots a_k$，那么就一定有 $0 = a_1 < a_2 < \ldots < a_k < n$，而这部分对答案的贡献是 $\prod_{i=1}^{k}(a_i + 1)$，所以总答案就是从 $[2, n]$ 中选出 $k-1$ 个数乘起来的所有方案的乘积之和。
分治 FFT 可以做到 $O(n \log^2 n)$。期望得分 80 分。

算法 5

辣鸡出题人卡我常数？

算法 5

辣鸡出题人卡我常数？
实际上有一种 $O(n\log n)$ 的做法。考虑你已经求出了 $F_t(x)=\prod_{i=1}^{2^t}(x+i+1)$，如何求 $F_{t+1}(x)=\prod_{i=1}^{2^{t+1}}(x+i+1)$？

算法 5

辣鸡出题人卡我常数？
实际上有一种 $O(n\log n)$ 的做法。考虑你已经求出了 $F_t(x)=\prod_{i=1}^{2^t}(x+i+1)$，如何求 $F_{t+1}(x)=\prod_{i=1}^{2^{t+1}}(x+i+1)$？
发现 $F_{t+1}(x)=F_t(x)F_t(x+2^t)$，后面的部分直接二项式定理展开，然后就只要做一次卷积而不用递归进右侧去计算了。

算法 5

辣鸡出题人卡我常数？
实际上有一种 $O(n\log n)$ 的做法。考虑你已经求出了 $F_t(x)=\prod_{i=1}^{2^t}(x+i+1)$，如何求 $F_{t+1}(x)=\prod_{i=1}^{2^{t+1}}(x+i+1)$？
发现 $F_{t+1}(x)=F_t(x)F_t(x+2^t)$，后面的部分直接二项式定理展开，然后就只要做一次卷积而不用递归进右侧去计算了。
复杂度 $T(n)=T(\frac{n}{2})+O(n\log n)=O(n\log n)$。
实测 $n,m=10^6$ 时跑得过（雾）。

算法 5

辣鸡出题人卡我常数？
实际上有一种 $O(n \log n)$ 的做法。考虑你已经求出了 $F_t(x) = \prod_{i=1}^{2^t}(x+i+1)$，如何求 $F_{t+1}(x) = \prod_{i=1}^{2^{t+1}}(x+i+1)$？
发现 $F_{t+1}(x) = F_t(x)F_t(x+2^t)$，后面的部分直接二项式定理展开，然后就只要做一次卷积而不用递归进右侧去计算了。
复杂度 $T(n) = T(\frac{n}{2}) + O(n \log n) = O(n \log n)$。
实测 $n, m = 10^6$ 时跑得过（雾）。
如果有更优做法欢迎来 D 出题人。

得分情况

内容…

吐槽环节

正解？

我做过 $ZJOI\ 2017$《树状数组》！

正解？

我做过 $ZJOI\ 2017$《树状数组》！
猜想给出的代码具有某种性质（如实际上求的是某个可以方便维护的值之类的）。

正解？

我做过 $ZJOI$ 2017《树状数组》！
猜想给出的代码具有某种性质（如实际上求的是某个可以方便维护的值之类的）。
可以获得 0 分的好成绩。

$n, q \leq 3000$

我会翻译代码!

$n, q \leq 3000$

我会翻译代码!
直接把给出的代码翻译成 $C++$ 即可获得 20 分。

$n, q \leq 3000$

我会翻译代码！
直接把给出的代码翻译成 $C++$ 即可获得 20 分。
我们简单理解一下给出的两段代码。第一段代码显然可以正确地资瓷单点修改和前缀查询两种操作，且单次修改复杂度 $O(k \log_k n)$，单次查询复杂度 $O(\log_k n)$。第二段代码单次查询的复杂度也是 $O(\log_k n)$ 的，单次修改复杂度 $O(n)$。

$n, q \leq 3000$

我会翻译代码!
直接把给出的代码翻译成 $C++$ 即可获得 20 分。
我们简单理解一下给出的两段代码。第一段代码显然可以正确地资瓷单点修改和前缀查询两种操作，且单次修改复杂度 $O(k\log_k n)$，单次查询复杂度 $O(\log_k n)$。第二段代码单次查询的复杂度也是 $O(\log_k n)$ 的，单次修改复杂度 $O(n)$。
所以这部分的复杂度大概是 $O(qn)$ 或 $O(qn\log_k n)$ 的，取决于你有没有预处理 $\mathrm{lowbit}(x)$。

$n, q \leq 2 \times 10^5, k = 2$

$n, q \leq 2 \times 10^5, k = 2$

诶这档分我怎么还没写就已经拿到了？

$n, q \le 2 \times 10^5, k = 2$

诶这档分我怎么还没写就已经拿到了？
显然在 $k = 2$ 时暴力的复杂度是 $O(q \log_2 n)$，因而可以获得这档部分分。

$n, q \le 2 \times 10^5$

关于这档分有一个正解无关的算法，但是很好写，所以就先讲了。

$n, q \leq 2 \times 10^5$

关于这档分有一个正解无关的算法，但是很好写，所以就先讲了。考虑将每个点 x 向 $x + \mathrm{lowbit}(x)$ 连边，如果后者超过了 n 就改为连向一个超级根，这样树状数组就变成了一棵有根树，而一次修改操作是把一个点到根路径上的每个点的权值异或上 v，查询是单点查询权值。

$n, q \le 2 \times 10^5$

关于这档分有一个正解无关的算法，但是很好写，所以就先讲了。考虑将每个点 x 向 $x + \mathrm{lowbit}(x)$ 连边，如果后者超过了 n 就改为连向一个超级根，这样树状数组就变成了一棵有根树，而一次修改操作是把一个点到根路径上的每个点的权值异或上 v，查询是单点查询权值。

直接用树状数组维护 dfs 序即可，这样单次修改复杂度 $O(\log_2 n)$，单次查询复杂度 $O(\log_2 n \log_k n)$，总时间复杂度 $O(n + q \log_2 n \log_k n)$。

$n, q \le 2 \times 10^5, 2 \nmid k$

还是考虑 x 向 $x + \mathrm{lowbit}(x)$ 连边这件事情。在 k 是奇数的情况下连出来的树是否存在什么特殊性质？

$n, q \leq 2 \times 10^5, 2 \nmid k$

还是考虑 x 向 $x + \mathrm{lowbit}(x)$ 连边这件事情。在 k 是奇数的情况下连出来的树是否存在什么特殊性质？

考虑 $x \leftarrow x + \mathrm{lowbit}(x)$ 这个操作的本质是什么，相当于是将 x 的最低非零位乘 2 并进位。可以发现在 k 是奇数时 x 的最低非零位永远不会变，因为 $2x \neq 0 \mod k$ 对于 $\forall x \in [1, k)$ 均成立。

$n, q \le 2 \times 10^5, 2 \nmid k$

还是考虑 x 向 $x + \mathrm{lowbit}(x)$ 连边这件事情。在 k 是奇数的情况下连出来的树是否存在什么特殊性质？

考虑 $x \leftarrow x + \mathrm{lowbit}(x)$ 这个操作的本质是什么，相当于是将 x 的最低非零位乘 2 并进位。可以发现在 k 是奇数时 x 的最低非零位永远不会变，因为 $2x \ne 0 \mod k$ 对于 $\forall x \in [1, k)$ 均成立。同理，由于模 k 意义下存在 2 的逆元，所以每个 x 都只会有至多一个后继。

$n, q \le 2 \times 10^5, 2 \nmid k$

还是考虑 x 向 $x + \text{lowbit}(x)$ 连边这件事情。在 k 是奇数的情况下连出来的树是否存在什么特殊性质？
考虑 $x \leftarrow x + \text{lowbit}(x)$ 这个操作的本质是什么，相当于是将 x 的最低非零位乘 2 并进位。可以发现在 k 是奇数时 x 的最低非零位永远不会变，因为 $2x \neq 0 \mod k$ 对于 $\forall x \in [1, k)$ 均成立。同理，由于模 k 意义下存在 2 的逆元，所以每个 x 都只会有至多一个后继。我们可以发现连出来的这棵树一定是根节点分叉的若干条链（或者如果不管根节点的话连成了若干条链），而每次操作都是给链的一段后缀异或上 v，所以对于每条链用个数据结构维护一下即可。复杂度 $O(n + q \log_2 n \log_k n)$。

$n, q \le 2 \times 10^5$

此时不能保证 $[1, n]$ 的所有点被连成若干条链。不过如果令 $k = 2^p \times t$，其中 t 是奇数，可以发现所有满足最低非零位上的值包含的 2 的因子个数不小于 p 的点会连成若干条链。

$n, q \le 2 \times 10^5$

此时不能保证 $[1, n]$ 的所有点被连成若干条链。不过如果令 $k = 2^p \times t$，其中 t 是奇数，可以发现所有满足最低非零位上的值包含的 2 的因子个数不小于 p 的点会连成若干条链。
只需要考虑剩下的点怎么维护答案就行了。

$n, q \le 2 \times 10^5$

此时不能保证 $[1, n]$ 的所有点被连成若干条链。不过如果令 $k = 2^p \times t$，其中 t 是奇数，可以发现所有满足最低非零位上的值包含的 2 的因子个数不小于 p 的点会连成若干条链。
只需要考虑剩下的点怎么维护答案就行了。
不太会？交个暴力试试？诶怎么过了？

$n, q \le 2 \times 10^5$

此时不能保证 $[1, n]$ 的所有点被连成若干条链。不过如果令 $k = 2^p \times t$，其中 t 是奇数，可以发现所有满足最低非零位上的值包含的 2 的因子个数不小于 p 的点会连成若干条链。
只需要考虑剩下的点怎么维护答案就行了。
不太会？交个暴力试试？诶怎么过了？
考虑每暴力走一步最低非零位值包含的 2 的因子个数就会 $+1$，那么在最低位不变的情况下最坏只需要暴力走 p 次（$\bmod k$ 不会使 2 的因子个数减少）。当然可能不幸的是你走着走着发现最低非零位乘 2 变成 0 了，这时候最低非零位就发生了改变，不过这样的改变至多只有 $\log_k n$ 次，所以一次暴力往上走的复杂度就是 $O(p \log_k n) = O(\log_2 k \log_k n) = O(\log_2 n)$。

$n \le 10^9, q \le 2 \times 10^5, k = 2$

树状数组用 `std::map<int,int>` 开就好了。
时间复杂度 $O(q \log_2^2 n)$。

$n \leq 10^9, q \leq 2 \times 10^5$

大下标可以使用离线离散化/动态开点线段树/平衡树解决。现在主要问题在于如何定位一个点 x 在哪一条链上，或者说，我们要找到 x 点所在链的链首。

$n \le 10^9, q \le 2 \times 10^5$

大下标可以使用离线离散化/动态开点线段树/平衡树解决。现在主要问题在于如何定位一个点 x 在哪一条链上，或者说，我们要找到 x 点所在链的链首。
考虑在 k 是奇数的时候哪些位置会成为链首。显然所有的 $(2x+1)k^y(0 \le x < \frac{k-1}{2}, y \ge 0)$ 会成为链首，且除此之外不存在其他的链首。

$n \leq 10^9, q \leq 2 \times 10^5$

大下标可以使用离线离散化/动态开点线段树/平衡树解决。现在主要问题在于如何定位一个点 x 在哪一条链上，或者说，我们要找到 x 点所在链的链首。
考虑在 k 是奇数的时候哪些位置会成为链首。显然所有的 $(2x+1)k^y(0 \leq x < \frac{k-1}{2}, y \geq 0)$ 会成为链首，且除此之外不存在其他的链首。k 是偶数时同理，所有数乘上 2^p 即可。

$n \le 10^9, q \le 2 \times 10^5$

大下标可以使用离线离散化/动态开点线段树/平衡树解决。现在主要问题在于如何定位一个点 x 在哪一条链上，或者说，我们要找到 x 点所在链的链首。

考虑在 k 是奇数的时候哪些位置会成为链首。显然所有的 $(2x+1)k^y(0 \le x < \frac{k-1}{2}, y \ge 0)$ 会成为链首，且除此之外不存在其他的链首。k 是偶数时同理，所有数乘上 2^p 即可。

之前讲过一条链上最低非零位是不会变化的，所以我们也就知道了最低非零位向上进位的次数（假设 x 的最低非零位是 q，那么进位次数就是 $\lfloor \frac{x}{k^{q+1}} \rfloor$）。

$n \leq 10^9, q \leq 2 \times 10^5$

大下标可以使用离线离散化/动态开点线段树/平衡树解决。现在主要问题在于如何定位一个点 x 在哪一条链上，或者说，我们要找到 x 点所在链的链首。

考虑在 k 是奇数的时候哪些位置会成为链首。显然所有的 $(2x+1)k^y(0 \leq x < \frac{k-1}{2}, y \geq 0)$ 会成为链首，且除此之外不存在其他的链首。k 是偶数时同理，所有数乘上 2^p 即可。

之前讲过一条链上最低非零位是不会变化的，所以我们也就知道了最低非零位向上进位的次数（假设 x 的最低非零位是 q，那么进位次数就是 $\lfloor \frac{x}{k^{q+1}} \rfloor$）。根据 x 最低非零位上的值以及进位次数，我们就可以倒推找到链首。

$n \le 10^9, q \le 2 \times 10^5$

大下标可以使用离线离散化/动态开点线段树/平衡树解决。现在主要问题在于如何定位一个点 x 在哪一条链上，或者说，我们要找到 x 点所在链的链首。

考虑在 k 是奇数的时候哪些位置会成为链首。显然所有的 $(2x+1)k^y(0 \le x < \frac{k-1}{2}, y \ge 0)$ 会成为链首，且除此之外不存在其他的链首。k 是偶数时同理，所有数乘上 2^p 即可。

之前讲过一条链上最低非零位是不会变化的，所以我们也就知道了最低非零位向上进位的次数（假设 x 的最低非零位是 q，那么进位次数就是 $\lfloor \frac{x}{k^{q+1}} \rfloor$）。根据 x 最低非零位上的值以及进位次数，我们就可以倒推找到链首。这个过程可以倍增优化至 $O(\log_2 n)$。

$n \leq 10^9, q \leq 2 \times 10^5$

大下标可以使用离线离散化/动态开点线段树/平衡树解决。现在主要问题在于如何定位一个点 x 在哪一条链上，或者说，我们要找到 x 点所在链的链首。

考虑在 k 是奇数的时候哪些位置会成为链首。显然所有的 $(2x+1)k^y(0 \leq x < \frac{k-1}{2}, y \geq 0)$ 会成为链首，且除此之外不存在其他的链首。k 是偶数时同理，所有数乘上 2^p 即可。

之前讲过一条链上最低非零位是不会变化的，所以我们也就知道了最低非零位向上进位的次数（假设 x 的最低非零位是 q，那么进位次数就是 $\lfloor \frac{x}{k^{q+1}} \rfloor$）。根据 x 最低非零位上的值以及进位次数，我们就可以倒推找到链首。这个过程可以倍增优化至 $O(\log_2 n)$。

因而本题的时间复杂度为 $O(k\log_2 n + q\log_2 n(\log_2 n + \log_k n))$，空间复杂度为 $O(q)$ 或 $O(q\log_2 n)$。

得分情况

内容…

吐槽环节

算法 1

`while(T--)puts("1");` 请！

期望得分 5 分。

算法 2

$n \leq 4$，大力手玩出所有情况，交个三维的表即可。
结合算法 1，期望得分 10 分。

算法 3

$n \leq 20$，大力状压判重即可。
结合算法 1，期望得分 20 分。

算法 3

$n \leq 20$，大力状压判重即可。
结合算法 1，期望得分 20 分。
好吧我承认这三档部分分都只是用来假装这题部分分多区分度良好而已。

算法 4

看见循环同构显然可以无脑套个 polya，于是答案就变成了：

$$\frac{\sum_{d|\gcd(n,m)} f(\frac{n}{d},\frac{m}{d})\varphi(d)}{n}$$

算法 4

看见循环同构显然可以无脑套个 polya，于是答案就变成了：

$$\frac{\sum_{d|\gcd(n,m)} f(\frac{n}{d},\frac{m}{d})\varphi(d)}{n}$$

$f(n,m)$ 表示什么？将 n 个珠子中的 m 个变成金的，要求最长连续段不超过 k，同时首尾连续段长度之和也不超过 k 的方案数。

算法 4

看见循环同构显然可以无脑套个 polya，于是答案就变成了：

$$\frac{\sum_{d\mid \gcd(n,m)} f(\frac{n}{d},\frac{m}{d})\varphi(d)}{n}$$

$f(n,m)$ 表示什么？将 n 个珠子中的 m 个变成金的，要求最长连续段不超过 k，同时首尾连续段长度之和也不超过 k 的方案数。考虑未变金的剩下 $n-m$ 个珠子，我们相当于要在 $n-m-1$ 个间隙以及首尾处中放入 m 个金珠子。

算法 4

看见循环同构显然可以无脑套个 polya，于是答案就变成了：

$$\frac{\sum_{d|\gcd(n,m)} f(\frac{n}{d},\frac{m}{d})\varphi(d)}{n}$$

$f(n,m)$ 表示什么？将 n 个珠子中的 m 个变成金的，要求最长连续段不超过 k，同时首尾连续段长度之和也不超过 k 的方案数。

考虑未变金的剩下 $n-m$ 个珠子，我们相当于要在 $n-m-1$ 个间隙以及首尾处中放入 m 个金珠子。易得其生成函数：

$$F(x)=(\sum_{i=0}^{k} x^i)^{n-m-1}(\sum_{i=0}^{k}(i+1)x^i)$$

答案就是 $[x^m]F(x)$。

算法 4

看见循环同构显然可以无脑套个 polya，于是答案就变成了：

$$\frac{\sum_{d|\gcd(n,m)} f(\frac{n}{d},\frac{m}{d})\varphi(d)}{n}$$

$f(n,m)$ 表示什么？将 n 个珠子中的 m 个变成金的，要求最长连续段不超过 k，同时首尾连续段长度之和也不超过 k 的方案数。考虑未变金的剩下 $n-m$ 个珠子，我们相当于要在 $n-m-1$ 个间隙以及首尾处中放入 m 个金珠子。易得其生成函数：

$$F(x)=(\sum_{i=0}^{k} x^i)^{n-m-1}(\sum_{i=0}^{k}(i+1)x^i)$$

答案就是 $[x^m]F(x)$。暴力多项式乘法 + 部分和优化可以做到 $O(n^2)$。

算法 $5: k=1$

$(1+x)^n=\sum_{i=0}^{n}\binom{n}{i}x^i$
最后一项变成了 $1+2x$，所以也可以直接算。

算法 $5: k = 1$

$(1+x)^n = \sum_{i=0}^{n} \binom{n}{i} x^i$

最后一项变成了 $1+2x$，所以也可以直接算。

$f(n,m) = \binom{n-m}{m} + \binom{n-m-1}{m-1}$

算法 6

我们考虑对 $F(x)$ 动一些手脚。

算法 6

我们考虑对 $F(x)$ 动一些手脚。
令

$$G(x)=\sum_{i=0}^{k}(i+1)x^i$$

则

$$G(x)x+\sum_{i=0}^{k+1}x^i=G(x)+(k+2)x^{k+1}$$

解得

$$G(x)=\frac{1+(k+1)x^{k+2}-(k+2)x^{k+1}}{(1-x)^2}$$

算法 6

于是就有

$$F(x)=\frac{(1-x^{k+1})^{n-m-1}}{(1-x)^{n-m+1}}(1+(k+1)x^{k+2}-(k+2)x^{k+1})$$

分式上面的部分用二项式定理展开后变成

$$\sum_{i=0}^{n-m-1}\binom{n-m-1}{i}(-x^{k+1})^i$$

下面的部分也可以用广义二项式定理展开变成

$$\sum_{i=0}^{\infty}\binom{a-b+i}{i}x^i$$

算法 6

于是就有

$$F(x)=\frac{(1-x^{k+1})^{n-m-1}}{(1-x)^{n-m+1}}(1+(k+1)x^{k+2}-(k+2)x^{k+1})$$

分式上面的部分用二项式定理展开后变成

$$\sum_{i=0}^{n-m-1}\binom{n-m-1}{i}(-x^{k+1})^i$$

下面的部分也可以用广义二项式定理展开变成

$$\sum_{i=0}^{\infty}\binom{a-b+i}{i}x^i$$

然后就可以算了？

算法 6

在计算 $f(n, m)$ 的值时，由于第三部分是三个单项式的和可以拆开计算，所以就只需要枚举第一部分 x 的指数。复杂度 $O(\frac{m}{k+1})$。

所以总体的复杂度就是 $O(\frac{\sigma_1(\gcd(n,m))}{k+1})$，由于 $\sigma_1(n)$ 大致可以认为是 $n \log\log n$ 级别，所以复杂度接近线性。

总结

本来这里应该有段总结的话的，但是它咕了。
附：在 `source/subtask` 文件夹下有出题人认认真真写的所有部分分程序供大家参阅。

谢谢大家！
祝诸君省选武运昌隆！