题解

zcysky

整数拆分

- Subtask1
- N<=100
- 直接搜索所有拆分方案

整数拆分

- Subtask2：
- N<=1000
- f(n)=f(n-1) (n%m!=0)
- f(n)=f(n-1)+f(n/m) (n%m=0)

整数拆分

- Subtask3:
- N<=100000
- 写个 mtt

整数拆分

- Subtask4
- K=1
- 可以用的数字一共有 1,m,m^2,m^3… 共 log 个。
- 令 f[i][j] 表示用了前 i 个数字，当前和为 j 的方案数。

整数拆分

- 令 j=k*m^(i-1)+r,r<m^(i-1)
- 注意到后面的都是前面的倍数，所以无论后面的数字如何选择，都不可能改变 r 的值，因此可能被用到的 j 的 一定是 k*m^(i-1)+(n%m^(i-1))
- 令 f[i][j] 表示用了前 i 个数字，当前数字和为 j*m^(i-1)+n%m^(i-1) 的方案。

整数拆分

- 这样 j 这一维还是太大。
- 通过观察和归纳证明，可以得到，对于确定的 i，f[i][j] 是一个关于 j 的 i 次多项式。
- 于是我们可以只保留前 i 项 dp 值，转移使用插值。
- 一共 log 个因数，每次需要做一次插值，log 次求值，复杂度 (logn)^3

整数拆分

- Subtask5
- M>10
- 留给一些插值带 log，或者听说有 (klogn)^5 之类的做法。

整数拆分

- Subtask6
- 其实这个卷积 k 次可以不用最后考虑。
- 卷积 k 次等价于，对于特定的值为 m^x 的数字，有 k 个不同的。
- 于是我们将 m^0…. 每个数字复制 k 次，直接做就行了。
- 最终要求的是前缀和，也很简单，直接再加一个值为 1 的数字就行了。
- 复杂度 (klogn)^3

简单计数

- 本题是留给其他两题做完了没事干的同学的。

简单计数

- Subtask1
- Sigma k*l<=1000
- 每次直接全部接好字符串，跑一次 manachar 即可。

简单计数

- Subtask2
- K=1
- 只要查询给定区间内部的回文子串数量。
- 经典问题，使用离线线段树或者可持久化线段树实现。
- 复杂度 nlogn

简单计数

- Subtask3
- 数据随机
- 注意到数据随机的情况下，回文串不会太长。
- 于是我们可以使用之前的方法查询区间内的回文串，跨过两个串边界的我们暴力跑几步来查询。

简单计数

- Subtask4
- K=2
- 主要是留给正解写挂的同学。
- 区间内部的回文串还是一样处理，考虑怎么处理跨过边界的回文串。

简单计数

- 令两个串分别为 s 和 t，
- 不失一般性，假设回文中心在 t 内。
- 如果回文串由长度为 x 的 s 的后缀，和长度为 y 的 t 的前缀构成，那么 t 长度为 y-x 的前缀显然必须也是回文串。
- 于是我们枚举每个 t 的回文前缀，求出以这个位置开始的串，和 rev(s) 的 lcp 即可。

简单计数

- 下面是几个简单结论。
- 一个回文串所有的回文后缀，可以由 logn 个等差数列表出。
- 如果l0,l1,l2…lm 是其中一个等差数列，令 d 为公差，那么 s(l0,l0+d-1)=s(l1,l1+d-1)=…=s(lm-1,lm-1+d-1)!=s(lm,lm+d-1)

简单计数

- 已知两个字符串 s 和 t，如果 s(0,d-1)=s(d,2d-1)=…=s((k-1)d,kd-1)!=s(kd,kd+d-1)，那么 t 和 s(0),s(d),s(2d),…s(kd) 的 lcp 一定是先不变，然后变化，然后以 d 递减。s(0)表示从 0 开始的后缀。
- 证明需要在纸上画画。

简单计数

- 有了这几个结论，我们就可以通过枚举每个等差数列，并且快速计算所有的 lcp 的和了。
- 回文中心在 s 中的情况，可以类比，只不过换成了枚举回文后缀。

简单计数

- Subtask5
- 其实如果两个串 s 和 t，如果知道了 st 的所有回文后缀，就可以做 k>2 了。
- 考虑 st 有哪些回文后缀，一部分是 t 本身的回文后缀，另一部分是跨过边界的，可以通过回文中心继续分成两部分。
- 考虑回文中心在 t 中的，可以发现它由一个 t 的回文前缀，以及足够长的 rev(s) 和t 之后的字符串的 lcp 组成的，这和求答案的过程几乎没有区别，因此我们只要在求答案的过程中判一下，lcp 是否足够长，如果足够长，就得到了新的回文后缀。
- 另外一种回文中心在 s 中的情况也是类似。

简单计数

- 剩下的问题就是如何求得一个区间的所有回文后缀，可以使用回文自动机来快速维护所有前缀以及后缀的回文后缀。

排序

- Subtask 1
- N<=100
- 随便怎么做都可以

排序

- Subtask2
- N<=1000
- 冒泡排序

排序

- Subtask3
- 值域 [0,5]
- 考虑 01 序列如何排序。
- 其实归并排序就可以，每次我们把左区间所有的 1 和右区间所有的 0，交换一下。
- 我们可以每次把一种数字分出来。
- 复杂度 6nlogn

排序

- Subtask4
- 值域较大
- 对所有数字再进行一次分治即可。