**cx觉得**

良心签到大水题。

从左到右把的所有字符压入栈中。加入一个字符之后，如果栈顶个字符形成的串和相等则把这个字符弹栈。结束之后栈中的所有字符连成的串即为答案。

。

**他的水平**

线段树优化DP经典题。

暴力的DP式子：

其中为就所有输入给出的，右端点不超过的区间产生的最大收益（且第块地必须不荒）。为的前缀和数组，为包含于的所有区间的收益之和，为的前缀最大值。

直接做是的，我们需要考虑优化。

首先，上面这个式子的可以移到外面，我们需要在不断递增的情况下，实时维护的最大值。

考虑使用线段树，第个位置的值即为。

一旦确定就不会再改变，故我们的重点在。

考虑加一的影响：显然，只比多了右端点为，左端点大于的所有区间收益之和。故需要计算时，先枚举所有右端点为的区间，对于一个这样的区间，对线段树的区间整体加上这个区间的收益即可。然后就是区间的最大值减去。的值计算好之后，将线段树的第个位置加上即可。

。

**比zzq**

定义为询问区间内所有值在之间的数之和。

首先，如果则直接返回内所有数的和。

否则由于，可以把递归到和。

考虑和线段树一样递归处理。易得，。

1. ：往递归。
2. ：往递归。
3. ：往两边递归。

我们会发现一个问题：递归下去的子问题变成了对或这样的序列进行查询，不满足的定义。

但我们注意到，区间内所有值在之间的数之和等于 区间内所有值在之间的数之和的倍，故对于递归下去的时候把和进行这样的处理即可。

对于，它的区间内所有值在之间的数之和等于 区间内所有值在之间的数的倍减一之和，也就是这些数之和的倍之和再减去这些数的个数。故实际上需要返回一个二元组，表示的区间内所有值在之间的数的个数与和。

。

**高**

简单倍增。

问题可以转化成枚举一个再枚举一个，将计入答案。

考虑把枚举改成枚举，那么这样的对答案的贡献就是。其中为到的路径上倒数第二个点。

不过由于的枚举范围是的严格祖先，在最坏情况下这样的复杂度还是的。

定义数组：

：的级祖先。

：的级祖先。

：

（为到的路径上倒数第二个点，为到的距离）

换句话说，表示已经确定，对于所有满足的，它们的之和。

和的转移比较简单：，。

边界，。

而对于，分两种情况考虑：

1. ，这一部分的贡献就是。
2. ，这一部分的贡献，除去第位的影响之外为。

现在考虑第位的影响。易得这就是所有满足的之和。即。

故我们得出了的转移：

最后答案即为。

。