题目大意

我们定义一个序列的权值为 $\max$(前缀最大值个数，后缀最大值个数)。

给定一个长度为 n 的**首尾相接**的排列 p，你需要将其分成 k 段非空序列，使得每个数都恰好在一段里，且 k 段序列的权值之和尽可能大。

数据范围

对于所有数据，满足 $1 \le k \le n \le 6 \times 10^5, 1 \le k \le 30$，保证输入的是一个 $\{1, 2, \cdots, n\}$ 的排列。

子任务编号	n	k	特殊性质	分值
1	≤ 30	-	-	10
2	-	$k = 1$	-	10
3	≤ 2000	-	数据随机	20
4	≤ 5000	-	-	20
5	$\le 5 \times 10^4$	-	-	20
6	-	-	-	20

数据随机：输入的排列在 $n!$ 种排列中随机生成。

解题过程

算法 1

枚举一个位置断环为链，预处理出每个区间的权值 $val_{l,r}$，令 $dp_{i,j}$ 表示前 i 个数被分成了 j 段的最大权值，有转移：

$$dp_{i,j} = \max_{1 \le k < i} dp_{k,j-1} + val_{k+1,i}$$

dp 时间复杂度 $O(n^2k)$，总复杂度为 $O(n^3k)$，可以通过子任务 1，期望得分 10 分。

算法 2

考虑 $k = 1$ 的情况怎么做。

注意到不管是前缀最大值还是后缀最大值，遇到 n 时都会被挡住，我们只需要求以 n 为某个端点的区间的权值即可，从 n 开始正反扫两遍，用单调栈维护最大值个数即可。

时间复杂度为 $O(n)$，结合算法 1 可以通过子任务 12，期望得分 20 分。

算法 3

数据随机的情况下，一段序列的前缀/后缀最大值个数期望为 $\log$ 级别，我们可以对枚举断点后的 dp 过程进行优化。

将前缀最大值个数和后缀最大值个数拆开，先考虑后缀最大值个数的转移，令 $sufval_{l,r}$ 为 $[l, r]$ 区间的后缀最大值个数：

$$dp_{i,j} = \max_{1 \le k < i} dp_{k,j-1} + sufval_{k+1,i}$$

注意到 $sufval_{k+1,i}$ 相同的 k 内我们只需要 $dp_{k,j-1}$ 最大的那一个，维护 $dp_{*,j}$ 的前缀最大值转移即可，前缀最大值个数的转移也是类似的。

dp 时间复杂度为 $O(nk\log n)$，总复杂度为 $O(n^2k\log n)$，期望得分 0 分。

算法 4

由于我们要把序列切成 k 段，会选取 k 个断点，每次随机选择一个断点断环为链进行 dp，期望只要选择 $\frac{n}{k}$ 次就能选到一个正确的断点，因此算法 3 复杂度可以优化为 $O(n^2\log n)$，结合算法 12，可以通过子任务 123，期望得分 40 分。

算法 5

我们考虑如何优化 $dp_{*,j-1}$ 转移到 $dp_{*,j}$ 的过程，定义两个辅助数组 A,B，$A_{t,s}$ 表示 $dp_{s,j-1}$ 转移到 $dp_{t,j}$，且最后一段是通过前缀最大值个数转移的权值，$B_{t,s}$ 表示最后一段是通过后缀最大值个数转移的权值，那么 $dp_{t,j}$ 的值就是 $\max_{s<t}\max(A_{t,s},B_{t,s})$。

考虑如何高效维护 A,B 数组的值。先观察 A_t 与 A_{t-1} 之间的区别：如果 $A_{t,s}\neq A_{t-1,s}$，一定是 p_t 被选入了前缀最大值中，这意味着 $[s,t]$ 之间没有比 p_t 大的数，这样的 s 一定是一段后缀，因此 A_t 就是 A_{t-1} 进行一次后缀加 1 后得到的数组。

再观察 B_t 与 B_{t-1} 之间的区别：首先 p_t 一定是后缀最大值，因此要先进行一次全局加 1。接下来考虑有一些数原先是后缀最大值，p_t 加入后不再是后缀最大值了，这样的数产生的贡献要删掉，即进行一次前缀减 1 操作。我们可以用单调栈来维护需要删掉哪些数，用两颗线段树分别维护 A,B 数组，dp 时间复杂度为 $O(nk\log n)$，总复杂度为 $O(n^2\log n)$，结合算法 2 可以通过子任务 123，期望得分 40 分。

算法 6

观察算法 5 中线段树所需的操作：

1. 后缀加 1
2. 前缀减 1
3. 后缀插入一个数
4. 求全局最大值

首先 2 操作可以看成全局 -1 后后缀加 1，因此可以忽略。

由于只有后缀加，那么如果有 $A_{t,x}\leq A_{t,y},x<y$。那么 $A_{t,x}$ 就不可能成为全局最大值了，我们可以用链表维护一个单调栈和它的差分数组，每次后缀加的时候找到加的位置在单调栈中的位置，剩余的部分就可以线性了，关键在于如何找到一次后缀加在单调栈中对应的位置。

我们用并查集维护每个点之后第一个在单调栈中的位置，初始 $fa_i=i$，在单调栈中删除一个数 p_i 时只需要将 i 和 $i+1$ 合并即可，dp 时间复杂度为 $O(nk\alpha(n))$，总时间复杂度为 $O(n^2\alpha(n))$，结合算法 2 可以通过子任务 1234，期望得分 60 分。

算法 7

我们重新审视一下算法 2 的做法：把问题转换为从 n 开始的链上问题。

这个做法能否拓展到一般情况呢？我们假设 n 旁边没有断点，包含 n 的那一段左右两侧一定有一侧是没有贡献的，假设这段选择了贡献前缀最大值个数，那么 n 之后的数就不可能成为前缀最大值了。这个问题可以看成断环为链后可以剩下一个后缀不被划分进 k 段内，只要枚举 2 个位置断开 dp 即可。

结合算法 5，时间复杂度为 $O(nk\log n)$，可以通过子任务 12345，期望得分 80 分。

结合算法 6，时间复杂度为 $O(nk\alpha(n))$，可以通过全部测试点，期望得分 100 分。