小 ω 的图

算法1

n 比较小的话直接搜索。可以拿到比较可观的分数。

算法2

如果只有5种不同的边权，那么只要枚举答案中有没有这个边权再看一下1 和 n 是否连通就好了。

算法3

$n = m$ 显然是一个基环树那么只有两条可能的最短路径，判断一下就好。

算法4

正解是从高位到低位贪心。如果当前位可以是 1，那么把所有符合条件的边拿出来，看一下 1 和 n 是不是连通就行了。

小 ω 的仙人掌

算法1

发现 $s \le 8$，可以使用暴力dfs枚举通过

复杂度 $O(2^n poly(n))$，期望得分 10 分。

算法2

$n \le 100$，暴力枚举 L 和 R，对 L 到 R 做背包，其中 f_i 为达成体积为 i 需要的最少代价，转移为

$f'_i = \min\left\{f_{i-A_j} + B_j\right\}$

询问只需判定 $f_w \le k$ 来确定 $R - L + 1$ 是否能更新到答案。

复杂度 $O(n^4)$，期望得分 30 分。

算法3

使用双指针（尺取法）。

发现只需要枚举 L 即可，右端点向右移动，同时插入元素，直到可以得到一个答案

这样就少了很多的重复插入

复杂度 $O(n^3)$，期望得分 $40 \sim 50$ 分。

算法4

对于算法3，给一个剪枝，如果当前枚举区间长度大于答案，就break掉。

复杂度 $O(n^3)$，期望得分 50 分。

算法5

对于测试点 6，因为 a_i 相同，所以取的物品个数是确定的。

判掉无解的情况，剩下的问题就变成了：

找到一个最短的区间，使得里面前 $\frac{w}{a_i}$ 大的和小于等于 k

因此很容易得出一个 $O(nw)$ 的做法。

其中可以使用算法 3 中的双指针，同时使用一个桶暴力得出前 K 大。

考虑判断存在性时是 $O(w)$ 的，当且仅当左端点移动或右端点移动时才会判断存在性，此时判断次数是 $O(n)$ 的。

复杂度 $O(nw)$，结合算法 4，期望得分 60 分。

算法6

考虑双指针的过程，实际上是维护一个队列的背包。

但是因为转移取的是$\min$，所以没有可减性。所以不能使用退背包。

考虑使用两个栈来模拟队列。

插入的时候在右边的栈顶放，弹出的时候在左边弹。

当左边弹空了，就把右边的栈倒过来插入到左边。

此时插入和删除复杂度都是 $O(w)$ 的。

此时可以 $O(w)$ 求出两个背包合并后单点的值，以判断存在性。

这样就可以不用利用可减性了。

根据算法 5 的复杂度分析，易得复杂度 $O(nw)$ 。

复杂度 $O(nw)$，期望得分 100 分。

小 ω 的树

算法1

首先我们可以得到一个$O(nlogn+nm)$的暴力做法。大概是把边按边权排序。从大到小插入后再看一所在联通块的点权和。这样就相当于求出了边权最小值为 v 时的最大点权和。于是对于所有这样的边都求一次就显然可以得到最优解了。

算法2

$x_i=1$那不是一个菊花吗？

那岂不是看中间的点必须选。

那么就是按边大小排序，然后就转变了区间a_i加，所有数$a_i\times b_i$的最大值。

这个显然就是一个大分块啊。每个块里搞一个一次函数极值就好了。

算法3

$u_i = 1$那岂不是每次只修改一个点。

那么如果他要选，就是按边大小排序，然后就转变了区间a_i加，所有数$a_i \times b_i$的最大值。

这个显然就是一个大分块啊。每个块里搞一个一次函数极值就好了。

如果他不选，剩下每个连通块求一遍就好了。

算法4

求出重构树。具体就是每次把两个连通块接在一起。用一个新的点代表这个联通块。

然后把这个点连向之前两个连通块。并且记录这个连通块的点权和a_i。边权最小值b_i

然后你单点修改，就是相当于修改了一个点到跟的子树a_i。

询问就相当于$max_{i=1}^{n} a_i \times b_i$

此时你使用树剖加分块，可以达到$O(nlogn + m\sqrt{n \log^3 n})$的优秀复杂度。但是没有足够的技巧你是无法通过的。

算法5

如果我们对于每一条重链，如果他长度为l，那么我们把它在序列上剖成$O(\sqrt{l})$个大小为$O(\sqrt{l})$的块。会不会优化复杂度？

首先我们发现这条重链上的时间复杂度为$O(\sqrt{l})$。

我们假设这个子树的大小为$size_x$

显然$\sqrt(l) <= size_x$，所以时间复杂度为$\sqrt{n} + \sqrt{\frac{n}{2}} + \sqrt{\frac{n}{4}} + ... = O(\sqrt{n})$

事实上根本跑不满。跑满貌似也是链的情况。

所以时间复杂度为$O(nlogn + m\sqrt{n}\log n)$。

大分块

前面多次提到了这个分块。

你需要实现对a_i进行区间加,求$a_i \times b_i$的最大值。

你对序列分块，大块维护李超树（李超树可能比较好写？），边角暴力并且重构，就可以了，时间复杂度$O(n\sqrt{n}\log n)$。