Solution

kill

小数据1

$dp[i][stat]$表示前i个人已经打了怪，$stat$压位记录哪些怪被打了，此时最多用时的人时间最少是多少。

复杂度$O(n^2 2^n)$。

小数据2

考虑二分答案，然后第i个人能不能打第j个怪物就确定了，跑二分图匹配看是否每个人都能找个怪物去打即可。

复杂度$O(n^3 log(Ans)$，其中Ans表示答案区间长度。

100%

法一：观察问题性质，我们可以首先得出，将人和怪按位置排序后，每个人打的怪物一定是有顺序的（第一个人打的怪物一定是所有被打怪物的第一只，第二个一定是第二只，等等），根据这个可以用$dp[i][j]$表示前i个人都已经打了怪物（并且是在前j个中间选择的），可以$O(1)$转移。

复杂度$O(n^2)$。

法二：我们还可以再发现另一个性质：打的怪物一定是一个连续的区间（很好证明，先自己尝试证一下）。所以我们排完序后，直接枚举第一个打的是哪只怪物即可。

复杂度$O(n^2)$。

（复杂度分析中，n, m为一个量级，故没有区分）

beauty

30%

暴力枚举配对方案。(每次枚举当前集合最小的数与哪个数配对)。

复杂度$O(n + (2k)!!)$，其中$n!! = n(n-2)(n-4)\cdots$。

50%

用$dp[S]$表示已经将S集合中的数完成了配对，它们的最大贡献是多少，每次枚举还没有选的那些点中最小的点与哪个还没有选的点匹配。

复杂度$O(n + (2k)2^{2k})$。

100%

我们用$dp[u]$表示走到u这个节点的边最多是多少条，从u走到它父亲的边是$min(dp[u], 2k - cnt[u])$，其中$cnt[u]$表示u这棵子树中的关键点数量，min的前面一项是u最多能提供的条数，后面一项是外面最多能接收的条数，两者较小值即为从u到其父亲最多的条数，我们只需要最后统计一下经过每条边的条数的和即可。

复杂度$O(n)$。

reverse

第1,2个点暴力模拟。

第3,4个点答案就是R。

其它情况考虑数位DP。

我们只需要算出[1,R]和[1,L-1]中满足reverse(n)也在[L,R]中的数的个数即可。

用$dp[i][j][s_1][s_2]$表示算了前i位，当前的前缀的长度都是j，并且将前缀reverse后与L和R的后j位比较结果为s_1, s_2，后面的选择有多少种（因为我是dfs写法，你们如果是递推写法，那么就表示前i位满足对应条件的前缀个数）（s_1, s_2只有三种取值，分别表示大于，小于和等于）。

如果a = 1，就不需要记录s_1，如果$b = 10^k$，就不需要记录s_2。状态会少一些。

复杂度是$O(T \times \text{状态数} \times 10)$。

weight

30%

因为$n = m$，所以是一个环套树，树边肯定会在最小生成树中，所以其答案为-1，环边想要在所有最小生成树中，当且仅当权值小于除它之外的所有环边的最大权值，故其余环边的最大权值-1即为该条边的答案。

复杂度$O(n)$。

另外30%

因为所有边的边权都是1，所以这道题变成了判断某条边是否一定出现在所有生成树中（即它是否为割边），如果是，那么它的答案为-1，否则它权值必须为0才能确保它在所有最小生成树中。最后用tarjan判一下割边即可。

复杂度O(n)。

100%

首先图的某一棵最小生成树求出来，对于树边和非树边分类讨论。

对于一条非树边，我们至少要将它的权值调整到树上这两个端点对应路径边权最大值-1才可以，否则我们一定可以不选这条边。显然，我们调整到这么大也足够了。

对于一条树边，我们关心的显然是两个端点对应的简单路径经过这条树边的那些边，我们最大的可能选择是那些边中权值最小的边的权值-1,（否则我们可以选那条最小边而不选这条树边），而我们如果将这条树边的边权调整成那个值，它也一定还会在最小生成树中。

所以这道题我们可以写一个树链剖分来完成我们上面的各种操作。

复杂度$O(nlog^2(n))$ 。