T1

首先 60 分是白送的。

第一步注意到 S_1 和 S_2 能匹配的串的长度有下界，就是非 `*` 的字符个数。

我们仔细观察一下这几个条件，可以得到几个性质：

- 由于 `*` 已经满足第一个条件，所以最多只有一个 `*` ，而没有 `*` 的情况出题人已经在数据范围里告诉了你。以下假设答案恰好有一个 `*` 。
- 由于 `*??????` 这样的东西可以匹配所有长度有下界的串，答案的串长应当恰好等于两个串中非 `*` 字符个数的较小值 $+1$ （因为还有一个 `*` ）。

有了这两个条件，其实要考虑的串已经很少了。我们可以直接枚举答案中 `*` 前的长度 l ，根据 S_1 和 S_2 的前 l 位可以确定 `*` 之前的结果，类似地也可以知道后若干位的结果，由于要求 `?` 的个数最少，这个串就唯一确定了。只需要把每种答案拿出来比较一下 `?` 的个数和字典序就可以做到 $O(n^2)$ 。接着注意到两个答案第一个不一样的地方一定是 `*` ，所以直接取 `?` 个数最少的中 l 最小的就行了，而 `?` 个数可以直接从左到右、从右到左推一遍得到，所以总的复杂度是 $O(n)$ 。这里注意确定前后两部分问号个数的时候要考虑 `*` 的必须全都是问号，比如 `*ak*` 和 `ak` 的答案是 `*??` ，而不是 `*ak` 。

T2

首先 30 分是白送的。为了严谨，这里不加证明地给出结论：不超过 T 的数有 $O(\log\log T)$ 个质因子，注意大 O 的定义是小于等于，不是严格等于。

令值域是 T 。一个简单的观察是：$r-l+1$ 一定是偶数，除非答案是 $(1,n)$ 。这是因为如果是奇数再加一个不会产生任何问题。我们可以先把每个数的质因子求出来，这可以做到 $O(T\log T)$ ，之后我们忽略这部分复杂度。求出了每个数的质因子后我们可以快速判断答案是否是 $(1,n)$ ，直接看是否有某个质因子出现了至少 $\frac{n}{2}$ 次即可。这样就得到了一个 $O(n^3\log\log T)$ 的算法，即直接枚举答案、暴力判断。

考虑先枚举质因子，判断是否有某个区间满足条件，可以转化为有某些位置是 1 ，其他位置是 -1 ，找到一个区间的和非负即为满足条件，即当 $pre_i \geq pre_j$ 的时候用 $(j+1,i)$ 去更新答案，那么把所有数按照 pre 排序之后就可以直接枚举 i ，最优的 j 是一个前缀最小值。这样做是 $O(n^2\log\log T\log n)$ 的。

为了进一步优化，我们要使得枚举每个质因子后的计算复杂度与 n 无关。为此，令 $p_1,...,p_k$ 为含有这个质因子的位置，从小到大排列。那么如果我们能找到 i,j 满足 $p_j-p_i+1\leq 2(j-i+1)$ ，那么就可以得到一个长度为 $2(j-i+1)$ 的答案，注意不一定以 p_i,p_j 为端点。另一方面，最优解一定会被这样考虑到。移项之后可以得到 $p_j-2j\leq p_i-2i+1$ ，所以我们把所有的数按照 p_i-2i 排序，枚举 j 之后 i 的最优解仍然是某个前缀最小值。这可以通过双指针或者二分求得对应位置。复杂度为 $O(n\log\log T\log n)$ 。最后，排序可以整体进行桶排，复杂度可以进一步降低到 $O(n\log\log T)$ ，不过这不是必须的。

如果用线段树进行这个操作，复杂度为 $O(n\log\log T\log n)$ ，不过常数比排序大得多，不一定能通过。

T3

首先注意到在相同的情况下 X 越小越好，所以 30 分是状压dp白送的。

为了解决这个问题需要一个重要的观察，在最优解中，一定有 Y 的前若干步增加的值都是 0 ，后面的都非零。这是因为如果某一步 Y 增加的是 0 ，那么可以直接把它往前提到第一步去，不会使其他的变劣。

容易证明最优解还需要满足的性质是在非爆零阶段一定是按照 a 从大到小排的，而爆零的部分显然怎么排都可以。到这里有多种方法都可能能解决部分分问题，比如枚举在哪一步开始不爆零、枚举爆零的 a 的和等等，复杂度大多与 $O(n^3X)$ 或 $O(nX^3)$ 类似。这里只讲导向正解的方法。

考虑枚举在爆零结束时的 X 值 x，那么我们按照 a 从大到小排序，依次决定每个位置是爆零还是不爆零，那么我们需要满足的是爆零的位置的 a 的和加上 x 要 $\geq X$ ，因为有可能 $x=0$ 。设计 $dp[i][p][q]$ 表示考虑了前 i 个，在不爆零阶段当前的 $X=p$ ，爆零阶段 a 的和是 q ，且当 $y>X$ 时候把 y 之间认为是 X 时当前最优的 Y 值是多少，这样做dp复杂度为 $O(nX^3)$ 。

为了优化，我们把这个dp的定义修改一下，p 的定义不变，q 的定义改为之后的数中爆零阶段 a 的和至少是 y 的时候这个dp值才能代表一个合法的方案。这样的好处是不需要枚举 x 了，直接把初值设为所有 $dp[0][x][X-x]=0$ 即可，最终的答案就是 $dp[n][p][0]$ 中的最大值，这样复杂度就降到了 $O(nX^2)$ 。

最后，注意到这个dp的转移方式：$dp[i][p][q]$ 转移到 $dp[i+1][\max(0,p-a_i)][q]$ 和 $dp[i+1][p][\max(0,q-a_i)]$ 很特殊，当 $p,q>0$ 时实际上有 $p+q=X-\sum_{j\leq i}a_j$ ，所以对于每个 i ，有效的 (p,q) 是 $O(X)$ 个，转移是 $O(1)$ 的，所以总复杂度是 $O(nX)$ 。