括号

$n \le 10$

对于这部分数据，通过 2^n 枚举每个串选或者不选，然后判是否是合法括号串即可。

把 `(` 看成 $+1$ ，`)` 看成 -1 ，那么一个括号串合法当且仅当前缀和始终 ≥ 0 且整个序列的和等于 0。

时间复杂度 $\mathcal{O}(2^n \times \sum |s_i|)$，期望得分 30 分。

$|s_i| = 1$

通过上述判合法括号串的方式，记 $f(i,j)$ 表示对于前 i 个串，当前前缀和为 j 的方案数。只需要保证转移过程中始终满足 $j \geq 0$ 即可。

初始 $f(0,0)=1$，转移有 $f(i,j)=f(i-1,j)+f(i-1,j-v)$，其中当 $s_i=$`(` 时，$v=1$ ，否则 $v=-1$ 。

最终答案为 $f(n,0)$。时间复杂度 $\mathcal{O}(n\sum |s_i|)$，期望得分 30 分。

正解

假设当前已经选择的串前缀和为 sum ，现在要选字符串 s，设 S 表示字符串 s 的前缀和，那么能转移当且仅当 $\min_{i=1}^{|s|}\{sum+S_i\} \geq 0$，即 $sum+\min\{S_i\} \geq 0$。

所以我们预处理 mn_i 表示 s_i 的最小前缀和，S_i 表示 s_i 的总和，那么转移式就有：

- 选择 s_i 加入，则 $f(i,j+S_i)+=f(i-1,j) \quad \text{if}(j+mn_i \geq 0)$;
- 不选择 s_i 加入，则 $f(i,j)+=f(i-1,j)$。

时间复杂度还是 $\mathcal{O}(n\sum |s_i|)$ ，期望得分 100 分。

路径

先考虑链的情况：对于 $a_i \leq 2$ 的情况。我们实质上就是看有没有一个 12 的子序列，这是容易 $O(n)-O(1)$ 处理的。

对于一般的链上情况：我们考虑 st 表：令 $k=\lceil \log_2 r-l+1 \rceil$：然后这个区间被划分成了左右两个部分。（这两个部分可能有重叠）。

如果 p_i,p_j 位于不同部分：比如 p_i 位于第一部分，p_j 位于第二部分减去两部分的并。那么我们发现 p_i,p_j 的选择区间是不交的，直接查询出两个区间的最小/最大的 a 即可。

如果 p_i,p_j 维护相同部分，那就是我们 st 表需要处理的事情：直接令 $f(i,k)$ 是 $[i,i+2^k-1]$ 的答案。那么 $f(i,k)$ 要么是 $f(i,k-1)$ 或者 $f(i+2^{k-1},k-1)$，要么就是在 $[i,i+2^{k-1})$ 和 $[i+2^{k-1},i+2^k)$ 两部分分别选出 p_i,p_j。

因此我们可以 $O(n\log n)$ 预处理，$O(1)$ 回答一次询问。

对于树上的返祖链，使用相同的手法：设 $f(i,k)$ 是 i 一直到 i 的 2^k-1 级祖先这一段的答案即可。

但是询问的时候，我们发现我们不再是 $O(1)$ 了，因为我们要支持求某个点的 k 级祖先。直接朴素做是 $O(\log n)$ 的，可以通过 $n,q\le 10^5$ 的部分。

比较熟知的 $O(1)$ 求 k 级祖先的方式是长链剖分。std 里的做法是把要求的所有内容离线下来。这样我们最后 dfs 一遍，维护当前的 dfs 栈，就可以离线均摊 $O(n+v)$ 求出 v 个 k 级祖先。

时间复杂度 $O(n\log n+q)$。

序列

$n=1$

发现若 $a<b$，那么答案为 $b-a$。否则就一直做 $a=\lfloor\frac{a}{2}\rfloor$ 直到 $a\le b$，然后加上 $b-a$。

因为若 a 加了若干个数后再除以二不如先除以二再加。

时间复杂度 $\mathcal{O}(T\log V)$，期望得分 10 分。

a_i,b_i 分别相等

其实和 $n=1$ 的答案是一样的。期望得分 20 分。

正解

我们把题意稍微改一下，当前有一个 $c=1$，可以执行以下两种操作：

- 令 $c=c\times 2$。
- 令 $a_i=a_i+c$。

设 $c=2^k$，2 操作加的总和为 S，那么最终有 $a_i'=\frac{a_i+S}{c}$。

令 $S_1=\lfloor\frac{S}{c}\rfloor,S_2=S\%c$，$a_{i,1}=\lfloor\frac{a_i}{c}\rfloor,a_{i,2}=a_i\%c$，那么有 $a_i'=S_1+a_{i,1}+\lfloor\frac{S_2+a_{i,2}}{c}\rfloor$。

可知此时的代价为 $S_1+\operatorname{popcount}(S_2)+k+\sum|a_i'-b_i|$.

而且最后一项 $\lfloor\frac{S_2+a_{i,2}}{c}\rfloor$ 只可能为 1 或 0。

不妨枚举 k，然后按照 $a_i\%2^k$ 排序。令 $a_i=\frac{a_i}{2^k}$，这样就可以通过操作 S_2 使得一个前缀的 a_i+1。假设枚举前 i 个，那么需要找到 $\min_{j=a_{i,2}}^{a_{i+1,2}-1}\{\operatorname{popcount}(j)\}$。这个通过分类讨论一下可以 $\mathcal{O}(1)$ 实现。

接下来我们需要确定 S_1，即问题转化为给定 a_i,b_i，每次可以让 a_i 全局 $+1$，最小化 操作次数 加上 $\sum|a_i-b_i|$。

这个问题等价于数轴上有 $n+1$ 个点，坐标分别是 b_i-a_i 和 0，选择一个非负轴的点使得所有点到这个点距离最小。显然取中位数即可。

现在复杂度的瓶颈在枚举前缀 a_i，复杂度为 $\mathcal{O}(n^2\log V)$。

发现其实就是单点修改，动态求中位数，随便用数据结构维护就行，时间复杂度 $\mathcal{O}(n\log^2 V)$ 或者 $\mathcal{O}(n\log V\log n)$。

不过上述做法还是常数略大。考虑到每次只会使得一个数轴上的点往左移一位，那么中位数也至多往左移一位，用 unordered_map 判一下就好了。期望得分 100 分。

其余部分分给一些比较暴力的实现和被卡常的正解。

回文

直接 m^n 暴力可以获得第一个子任务的得分。

看到这个题一个最直接的想法就是对于一个确定的 n 去算 $\sum_{i=0}^{i-1} m^{(\lceil \frac{i}{2} \rceil + \lceil \frac{n-i}{2} \rceil)}$，但是一个串可能会有多种 合法的拆分，例如 aaa。

因此我们来研究一个串如果有两种合法拆分（回文也算一种拆分）会有什么性质，通过一些打表观察可 以得到结论：若一个串有两种合法拆分，则其一定是有一个本原合法串复制若干遍组成。（这里，本原的意思是，其拆分是唯一的）

因此我们设上面的式子是 $f(n)$，而 $g(n)$ 是长度为 n 的本原串个数，则有：

$$g(n) = f(n) - \sum_{d|n, d<n} g(d) \times \frac{n}{d}$$

如果计算出 $f(d)$，则可以 $O(n \ln n)$ 计算 g。而 f 容易在 $O(n \log n)$ 的时间求。这样就能通过子任务 2。我们还需要更快的做法。

首先根据莫比乌斯反演可以得到：

$$g(n) = \sum_{m|n} f(m) \times \mu(\frac{n}{m}) \times \frac{n}{m}$$

令 g 的前缀和函数为 G，f 的为 F；令 $\mu'(x) := \mu(x) \times x$。

首先 $F(n)$ 可以利用错位相减 $O(\log n)$ 求出。这里需要特判 $m = 1$。

然后 μ' 的前缀和 M 也是能快速求的：注意到 μ' 和 $I(x) = x$ 的卷积结果就是 $\epsilon(n) = [n = 1]$，因此可以杜教筛在 $O(n^{\frac{2}{3}})$ 的时间内求出每个 $\lfloor \frac{n}{x} \rfloor$ 处的点值。

求出 F, M 后，利用整除分块即可求出：

$$G(n) = \sum_{d} f(d) \times \mathrm{M}(\lfloor \frac{n}{d} \rfloor)$$

而我们要求的是：

$$\sum_{d} g(d) \times \lfloor \frac{n}{d} \rfloor$$

直接用整除分块套整除分块是 $n^{\frac{3}{4}}$ 的，可以通过子任务 3；如果常数优秀或许也能满分。

实际上我们类似杜教筛：预处理 $n^{\frac{2}{3}}$ 以内的 G 的点值，即可 $O(n^{\frac{2}{3}})$ 解决本题。