Solution

bike

我们记$T_n=\overline{S_n}$，然后有：

$$S_n=S_{n-1}T_{n-1}$$
$$T_n=T_{n-1}S_{n-1}$$

我们用$f[u][p][0/1]$ 来表示“从u号点开始走，走一个S_p（或T_p如果第三维为1）的路径后到达的点集”（即f是一个点集，可以压位实现）。类似的，用$g[u][p][0/1]$来表示另一个点集，对于该点集的点，都存在一条从该点出发，走一个S_p（同样，根据第三维是0还是1来决定是S_p还是T_p）的路径，能够到达u。 这样我们就可以用f和g相互递推（$f[u][p-1][0]$和$g[v][p-1][1]$如果有交集，则说明从u到某个点走S_{p-1}，从那个点到v走T_{p-1}，即存在一条路从u到v路径是S_p）

递推完了后，我们用$dp[u][p][2]$表示“从u开始走，第一步步长最多为S_p（或T_p如果第三位为1），最远能走多长。这个可以用f来动归。

最后，枚举最长的第一步是多长，即可。

其中f和g可以手写压位（用unsigned long long手写），也可以用bitset。

（建议参考标程代码阅读上面的题解）

contest

我们根据Landau's 判别法，可以发现：

$$30n\geq\frac{(n-1)n}{2}$$

从而解出：

$$n\leq 61$$

显然，我们先枚举n有多大，然后做一次dp来判断是否存在方案。

我们本质上要做的事情是，判断是否存在一种方案，将原来的每个a多复制几次，最终达到有n个数，并且满足判别法所要求的条件。

我们用：$dp[i][j][s]$表示，我现在一共有了i个数，且最后一个数是a_j，所有数的和是s，并且前i个数满足判别法，这种方案是否存在（即dp存的值是boolean）。每次的转移有两种：

- 第$i+1$个数还是选择a_j。
- 第$i+1$个数选择a_{j+1}。

最后，如果$dp[n][m][C_n^2]$为True，则说明这个n是合法的。

考虑怎么输出方案，显然，我们需要记录dp状态的转移过程（即每个状态的前驱状态是什么），然后根据这个信息计算出每个数分别出现了多少次。假设我现在得出了每个人获胜的次数分别是：

$$d_1,d_2,d_3,\dots,d_n$$

将他们从小到大排序，然后将d_1对应的人赢$d_2, d_3, ..., d_{d_1+1}$这些人，输给其他人，然后递归构造（我们考虑，我们的子问题的d是满足判别法的，故一定有方案，故这种构造一定可以构造出解）。

（同样，可以参考代码理解上面的内容）

diameter

这道题，假如我们一开始回答了(u, v)的询问，就把它的答案保存下来，下次回答的时候直接输出。

然后我们每次询问如果用$O(较小的连通块大小)$这个复杂度解决，那么我们就可以用$O(q\frac{n}{\sqrt{q}}))$的时间复杂度解决（最坏情况下，每次询问的都是不同的连通块对，并且每个连通块对的大小都相同，从而有上面的复杂度）。

考虑怎么用上面的"较小的连通块大小"的复杂度解决，对于两个连通块，首先预处理出每个点到所在连通块的最远点的距离（d_u），假如连接的是u和v两个点，则新的树的直径是（其中D_1, D_2是原来连通块的直径）：

$$max(D_1, D_2, d_u + d_v + 1)$$

如果我们可以枚举u，那么哪些v对应的贡献是$max(D_1, D_2)$，哪些v对应的贡献是$d_u + d_v + 1$就可以快速的算出来（中间可能需要一个二分，所以实际的复杂度应该是$O(alog(b))$，其中a是较小的连通块大小，b是较大的。

即使多个log，复杂度也是可以接受的。

（注意，期望值显然是：

$$E = \frac{\sum_{u \in U, v \in V} max(D_U, D_V, d_u + d_v + 1)}{size[U]size[V]}$$

)