T1 -- 小组作业(group.cpp)

解法：

定位为签到题。

把所有人从小到大排序，取出最大的 k 个人 b_i ，剩下的 $n-k$ 个人可以当这 k 个人中的任意一个人使用。

考虑二分答案 mid ，判定一下 $\sum\max(mid-b_i,0)$ 和剩下 $n-k$ 个人权值之和谁大谁小，时间复杂度为 $n\log n$。

T2 -- 排列(permutation.cpp)

解法：

能够证明逆序对个数不会超过 200 个。

$f[i][j]$ 表示长度为 i 的排列中，逆序对个数为 j 的排列有多少个，可以利用 DP + 前缀和在 $O(200n)$ 的时间复杂度内计算。

根据 $f[i][j]$ 可以计算出排名第 k 小的排列有多少个逆序对。

然后从第一位逐位开始确定，假设已经确定了前 x 位，我们可以知道前 x 个数字内部之间的逆序对个数，和前x 个和后 $n-x$ 个数字之间的逆序对个数，从而推出后 $n-x$ 个数字内部之间的逆序对个数，这个方案数可以利用$f[i][j]$ 快速计算出，时间复杂度为 $O(200n)$，如果实现的不精细会退化到 $O(n^2)$。

T3 -- 四舍五入(round.cpp)

解法：

考虑如何判定 x 能否变成 y。

从低位到高位逐位比较，若它们当前位相同，则不进行进位，否则只有当 y 这一位为 0 或 1 时才可能有解。

具体原因为对 x 当前位四舍五入后，该位会变成 0，变成 0 以后，前面的位如果进位，那么当前位会变成 1，显然一个位置不可能进位 2 次，因此只能得到 $0, 1$ 两种值中的一种。

注意到以上性质后，我们有了一个 DP 的雏形，$f[i][0/1/2]$ 表示当前考虑到了第 i 位，上一位不会进位（0），必须进位（1），可以进位也可以不进位（2），进行数位 DP 的方案数。

注意，如果只考虑了 $0, 1$ 状态而没有单独分类 2 状态是会记重的。

具体的进位与否需要手动进行讨论，具体为以下几种情况：

> 1. $y = 0, 0 \leq x \leq 3$ 无论前面是否进位，当前位一定不进位。
> 2. $y = 0, x = 4$ 如果前面进位，那么当前位可进位可不进位，否则当前位不进位。
> 3. $y = 0, 4 < x \leq 9$ 无论前面是否进位，当前位一定进位。
> 4. $y = 1, 0 \leq x \leq 4$ 一定先四舍五入当前位，前面那一位再进上来，因此当前一定不进位，前面能进位。
> 5. $y = 1, 5 \leq x \leq 9$ 一定先四舍五入当前位，前面那一位再进上来，因此当前一定进位，前面能进位。

复杂度为 $O(10T \log V)$。

T4 --比大小 (compare.cpp)

解法：

本题的关键性质为，答案一定是形如$AAAABBBBCCCCDDDDEEEE...$+ T 的一个完整后缀。

如果出现了 $AAAABBBBCCCCB...$ 这样的结构，如果以 B 为开头不是一个后缀，那么可以修改为 $AAAABBBBCCCB...$ 因为不是后缀，所以可以在后面再多取一个字符。

观察到这个关键性质以后，注意到字符集大小为 26，因此我们可以先求解出前缀有多少个 A，具体方法为二分（或者直接预处理，实现的更精细可以做到 $26n$，时限已经调大，两种方式预估都能通过）进行判定，然后再看接下来可以接多少个 B ，多少个 C，多少个 D，以此类推，求解出 $f(S[l_i, r_i], k)$ 每个字符分别有多少个，以及最后的后缀的形态。

比较大小时，可以通过二分+ hash 的方式进行比较，实现的精细的话单次复杂度为 $O(26 + \log n)$，不精细复杂度为 $O(26 \log n)$，两种方式预估都能通过。

最终复杂度为 $O(26n + q \log n)$ 或 $O(26n + q26 \log n)$。