因为有些 ddl 要肝.. 把视频题解咕了..大家凑合着看文字题解吧

今年又是 Rikka 和 Yuta 在多校上秀恩爱的一年（说不定也是最后一年了.. sad

1001 Rikka with Quicksort

难度：medium

考察了大家对快速排序时间复杂度推导的熟悉程度（当然我是打表推式子的）

如果打表的话可以考虑下面这个式子，它是计算在 m 处增加一个 1 对答案的贡献。

$$
\begin{aligned}
h_m(i) &= 0 && i < m \\
h_m(i) &= 1 && i = m \\
h_m(i) &= \frac{1}{i}\sum_{j=1}^{i}\left(h_m(j-1)+h_m(i-j)\right) && i > m
\end{aligned}
$$

打表之后可以知道 $h_m(m)=1, h_m(n)=\frac{2(n+1)}{(m+1)(m+2)}(n>m)$。应该归纳一下就可以直接证明。所以把对应的贡献求和之后可以得到答案就是：

$$
2(n+1)H(n)-4n-\frac{2(m+2)H(m+1)-4(m+1)-m)(n+1)}{m+2}
$$

其中 $H(n)$ 表示调和级数。因为模数确定，可以用分段打表的策略来求调和级数的具体值：即把 $H(kS)$ 的值给打入程序中，这样每一次只需要计算 $kS+1$ 到 n 的逆元就行了。

1002 Rikka with Cake

难度：medium easy

用欧拉（欧拉欧拉欧拉）公式 $V-E+F=2$ 来算蛋糕的块数。

首先考虑点数 V，蛋糕的四个角以及射线的端点贡献了 $n+4$ 个顶点（K 太丑了我们用 n 表示射线数），射线和四条边的交点共有 n 个，射线之间的交点有 c 个，因此点数是 $2n+4+c$。

接着考虑边数，对于每一条射线，假设别的射线和他的交点有 c_i 个，那么这条射线被切成了 c_i+1 段，因此所有射线的边数对应的是 $2c+n$（因为每一个交点被用了两次）。同时因为四条边一共和射线交了 n 次，所以四条边界上共有 $4+n$ 条边，所以 $E=2c+n+4$。

因此总的区域数为 $F=2+c$，因为要去掉最外面的无穷区域，所以答案就是 $1+c$。于是问题就变成了求交点个数 c。这是一个经典的问题，分四个方向讨论一下离散化用树状数组求就行。时间复杂度 $O(n\log n)$

1003 Rikka with Mista

难度：medium-easy

在，为什么迫害米斯达?

可以对每一位分开来计算，和最大不超过 10^{10}，因此最多只有 10 位。只要对每一位算出它在多少种情况下是 4，全部加起来就可以了。

40 的数据范围指明了可以用 meet in middle 来做，可以先用 $O(2^{\frac{n}{2}})$ 的复杂度分别把前 $\frac{n}{2}$ 个数和后 $\frac{n}{2}$ 个数的所有和求出来，分别存在数组 A 和 B 中，那么对于第 i 位来说，答案就是有多少个数对 i,j 满足 $(A_i+B_j)\bmod 10^{i+1}\in[4\times 10^i,5\times 10^i)$。这个问题只要对 A 和 B 按照 $x\bmod 10^{i+1}$ 排序后用 two pointer 扫就可以了。

对每一位分开来做时间复杂度是 $O(2^{\frac{n}{2}} n \log w_i)$，瓶颈在堆每一位排序。不难发现对每一位排序可以用一次归并排序来实现，这样就能省下一个 $\log$，时间复杂度是 $O(2^{\frac{n}{2}} n)$。

1004 Rikka with Geometric Sequence

难度：medium

首先长度是 1 的答案就是 n，长度为 2 的答案就是 $\frac{n(n-1)}{2}$。我们考虑长度大于等于 3 的情况。

假设公比的最简分数表示是 $\frac{a}{b}(a>b, \gcd(a,b)=1)$，等比数列长度是 k，那么 x 能作为这样的一个等比数列的最后一项的条件就是 $a^{k-1}|x$，因此这样的等比数列一共有 $\left\lfloor \frac{n}{a^{k-1}} \right\rfloor$。而枚举了 a，满足小于 a 且与 a 互质的 b 的数量是 $\varphi(a)$，其中 φ 表示欧拉函数。 因此我们可以枚举 a 然后对答案求和：

$$\sum_{a=2}^{n} \varphi(a) \left\lfloor \frac{n}{a^{k-1}} \right\rfloor$$

显然有贡献的 a 必须小于等于 $\sqrt[k-1]{n}$，因此当 $k>3$ 的时候就能暴力枚举 a 统计答案了，问题就在于 $k=3$ 的时候的处理。这个时候可以证明 $\left\lfloor \frac{n}{a^2} \right\rfloor$ 的取值最多只有 $\sqrt[3]{n}$ 种：当 $a \le \sqrt[3]{n}$ 的时候，只有 $\sqrt[3]{n}$ 种值，当 $a > \sqrt[3]{n}$ 的时候，这个商小于等于 $\sqrt[3]{n}$，因此最多也就只有 $\sqrt[3]{n}$ 种答案。

所以相当于把 a 的取值分成了 $\sqrt[3]{n}$ 段，对每一段分别用杜教筛求 $\varphi(n)$ 的前缀和就可以了。这样的时间复杂度比较迷幻，我展开一层积分一下可以得到时间复杂度是 $O(n^{\frac{5}{12}})$。不确定最终的时间复杂度是什么。但是如果预处理前 5×10^7 项的欧拉函数，可以非常轻松的跑过去。

1005 Rikka with Game

难度：easy

签到题，随便脑补一下就能知道答案。首先如果第一位是 $a-x$ 的话，先手会直接结束游戏：如果先手增加第一位，那么后手直接结束游戏；如果先手不增加第一位，那么后手增加第一位，此时如果先手还不结束游戏，后手直接选择结束。这三种情况下游戏结束的字典序都要比游戏开始的时候字典序大。

如果第一位是 y，那么不管先手还是后手都不会选择修改它：如果先手修改它，后手直接结束游戏；如果后手修改它，先手再选择同一位，然后在下一次轮到先手的时候结束。这样对于修改 y 的那一方都是亏的。

如果第一位是 z，那么先手会选择修改它，之后后手也会选择修改它，然后就回到了第一种情况先手直接结束游戏，因此结果就是第一位变成了 b.

在上面三种情况下，只有第二种情况游戏不会再只考虑第一位的情况下结束，这时因为大家都不会动第一位，所以相当于不存在，直接考虑后面的情况就可以。

所以最后的结论就是，找到第一位不是 y 的位，如果它是 z，则修改成 b，否则不变。 时间复杂度 $O(n)$。

1006 Rikka with Coin

难度：easy

首先 10 分的硬币最多只会用一个，如果用了两个，直接替换成一个 10 分一个 20 分一定不亏。

20 分的硬币最多只会用三个，如果用了四个，直接替换成一个 10 分两个 20 分一个 50 分一定不亏。

50 分的硬币最多只会用一个，如果用了两个，直接替换成一个 50 分和一个一元一定不亏。

对于任何一种情况，重复使用上述规则一定会达到一个 10 分硬币最多一个，20 分最多三个，50 分最多一个的情况，不会陷入重复甩锅的死循环。

因此枚举这三种硬币分别用了多少个，然后整百的部分直接用一元硬币填，取最少的答案就行了。

1007 Rikka with Travel

难度：medium easy

基础直径练习题。

考虑判断 (x, y) 能不能出现。劼论：任意取树上的一条直径，那么如果 (x, y) 能出现，那么一定存在一种方案使得直径的两端都被使用了。证明很简单：假设存在一个端点没有被使用，那么考虑两条直线的四个端点 a, b, c, d，一定可以把一个端点给移动到直径的这个端点上，因为直径是树上最长的路径，因此这次移动一定不会减少路径的长度。

考虑对每一个长度 x，求可以满足的最长的 y，这样所有小于等于 y 的值也都能被满足。考虑在直径上计算这些值：第一种情况，直径的两端分属不同的路径，那么可以枚举直径的一个端点那条路径在直径上的最后一个点，那么这一条路径的最大长度就是这个点到直径端点的距离加上这个点往直径外的最大延伸长度，另一条路径的最大长度也可以类似地求，其中最大延伸长度可以用一个 $O(n)$ 的 DFS 计算出来。第二种情况是有一条路径就是这条直径，那么另一条路径就是这条直径之外的最长路径长度，这个只要把直径上的点都删了再求一遍直径就行。

总的时间复杂度为 $O(n)$。

1008 Rikka with Stable Marriage

难度：medium

稳定婚姻真实一个深刻的模型啊..这题明明很简单但是不知道为什么没人做。

考虑把男生和女生分别放到一棵 trie 树里面，然后考虑最高位：首先肯定是让不同子树的男生女生进行匹配，如果左子树里有一队，右子树里有一队，那么交换他们的伴侣肯定一个更好的选择。而因为左右子树的大小可能不均匀，所以在分开来匹配之后，不妨设左子树有一些男生剩下，右子树有一些女生剩下，这个时候就只能让败者组的他们互相舔舐伤口了。

写成代码很简单，下面是一个简单的伪代码：

```
void match(node_b, node_g, depth) {
    if (node_b.size == 0 || node_g.size == 0) return;
    if (depth == -1) {
        marriage(node_b, node_g); return;
    }
    match(node_b.l, node_g.r, depth - 1);
    match(node_b.r, node_g.l, depth - 1);
    match(node_b.l, node_g.l, depth - 1);
    match(node_b.r, node_g.r, depth - 1);
    node_b.update_size();
    node_g.update_size();
}
```

其中 `node_b` 和 `node_g` 分别表示男孩和女孩对应的 trie 树节点，`depth` 表示现在考虑的是哪一层。大致的思想就是暴力递归下去然后贪心的匹配就行了。

因为每递归 $O(\log a_i)$ 层一定会有一对新人产生，所以总的时间复杂度是 $O(n \log a_i)$ 的。

1009 Rikka with Traffic Light

难度：medium hard

首先可以发现一定存在一个最优方案，所有绿灯区间长度都大于等于 T_1，所有红灯区间长度都大于等于 T_2。如果如果小于的话，反正这一段时间也没有办法让人过红绿灯，不如就不变了。接着考虑如果灯在第 t 时刻变成了绿灯，最优解中它会在什么时候变成红灯。（对于红灯变成绿灯的分析类似）

第一种可能性是在 $t+T_1$ 时刻变成红灯，在这种情况下，一定有在 t 之前（包括 t） 时刻到达且还没有过马路的人，不然这一段没有人能过马路。

第二种可能性是在 $t+T_1$ 时刻之后，考虑在这段绿灯中最后过马路的那个人 i，他会在 t_i+T_1 时刻后通过马路。因此这段绿灯在第 t_i+T_1 时刻结束一定不亏。

我们把所有 t_i+T_1 或者 t_i+T_2 (取决于 i 的种类) 作为关键点，设 f_i 为在第 i 个人对应的关键切换成另一种灯时的已经能确定过马路时间的人的总等待时间（不妨假设第 i 个人是第一类人）。注意这儿已经能确定过马路时间的人包括在 t_i 时刻之前到达的第一类人和第 t_i+T_1 时刻之前达到的第二类人 (最开始发现的性质保证了他们都能在 t_i+T_1) 时刻过马路。

考虑转移，首先从 t_i+T1 开始，最优方案可能会进行若干段第一种可能性的转移，接着通过一个第二种可能性的转移直接跳到后面的某一个关键点。因为第一种可能性要求必须要有对应的人在等待，因此第一种转移最多进行 $O(n)$ 次。我们可以枚举第一段转移进行的次数，并求出对应的总等待时间（总等待时间的定义和 f_i 一样）。

最后要处理的就是从枚举的这 $O(n)$ 段到后面的某一个关键点之间的转移，这个把式子列出来之后可以发现用斜率优化就能直接优化到 $O(n)$ 了。

总时间复杂度为 $O(n^2)$。数据造的我想吐。

1010 Rikka with Defensive Line

难度：medium hard

我九条可怜最喜欢做的一件事，就是出大几何题，对自以为多开就能 AK 的人说“不”。

过气计算几何选手拼尽全力出的计算几何题..然后根本没人帮我验题..偷偷搞了一个 60 核 server 并行用极角排序验了一下.. 发现这个数据范围的 $O(n^2\log n)$ 的极角排序并行后两三分钟就跑完了.. 感觉有点牛逼

首先可以发现对于一个凸包来说，如果一个直线和它有交，那么一定在直线的两侧（不严格）都有至少一个凸包上的顶点。因此我们对点集剥 m （因为 K 太丑这儿就用 m 代替了）层凸包（重复求凸包，删除凸包上的点这一过程 m 次），只有这些凸包上的点可能作为防御线的两个点之一，其他点都没有用了。

枚举一个点集中的点，把这个点作为坐标系的原点，那么对于任何一个方案，它小于等于 m 的那一边一定覆盖了 y 轴正半轴或者负半轴中的一个。可以先求所有覆盖了 y 轴正半轴的方案，那么另一边只要对称一下再做一遍就行。同时不妨假设另一个端点在 x 轴正方向，逆方向的情况也只要对称一下就行。

考虑这时哪些点可能作为答案：答案是把 y 轴沿着逆时针防线方向旋转碰到的前 m 个点。同时为了判断这些点是否合法还需要求沿着顺时针方向旋转碰到的前 m 个点。如果把这些点求出来，那么就只要对每一个点看一下这 $2m+1$ 个点（包括原点）里是否有超过 m 个点在这一边就行，这是一个非常基础的 two pointer （注意处理共线的情况）。

问题的关键就是如何求逆时针方向（顺时针方向同理）的前 m 个点。因为现在的点集有特殊性：由 m 个相互嵌套的凸包组成，因此我们考虑只有一个凸包的特殊情况。讨论可以发现，y 轴正半轴和凸包的交点以及原点到凸包的切点把凸包分成了若干段，其中每一段从极角上考虑都是有序的。因此 m 个凸包一共分成了 $O(m)$ 段，用一个多路归并把这些段归并起来就行。求交点可以按照 x 轴排序枚举原点，这样就可以分上下凸壳分别扫过去；求割点可以用旋转卡壳预处理，这两部分总的时间复杂度都是 $O(nm)$ 的。

算上 m 路归并的复杂度 $O(m\log m)$，总的时间复杂度为 $O(nm\log m)$。

1011 Rikka with Segment Tree

难度：medium

这个题是一个简单的分形题.. 做法也很传统，就是大力递归下去，难点在于比较麻烦。

我们用 $F(N)$ 表示到线段树的区间长度到 N 为止的求和。那么除了 1 的情况直接结束，其他的线段树都会递归左子树和右子树。如果我们假设左右子树都是从 1 开始重新标号的话，那么左右子树都是递归到 $\frac{N}{2}$ 上下取整的子问题，这一部分用一个记忆化的递归就行。

考虑合并左右子树成 n 的情况，首先所有叶子节点的深度都会增加 1，这一部分的贡献是 $\sum_{n=2}^{N}\sum_{i=1}^{n} i$，可以直接计算。

最后，因为加上左子树之后，右子树的所有点的编号会对应地发生变化，因此需要再算上对应的贡献。令 $g(n)$ 表示对应区间为 n 的线段树所有叶子节点的深度和，设 m 是最大的整数满足 $2^m \leq n$，那么 $g(n) = nm + 2(n - 2^m)$。而这一部分的贡献就是：

$$\sum_{n=2}^{N} \left\lceil \frac{N}{2} \right\rceil g\left(\left\lfloor \frac{n}{2} \right\rfloor\right)$$

推一个式子加上一通预处理，就能 $O(1)$ 计算这个和式了。

如果用 map 来进行记忆化的话，时间复杂度为 $O(\log^2 n)$。