A. Wiki with Horses

算法：简单贪心

非常简单的贪心，建立一个结构体变量，里面包含两个参数 t 和 a 。先拿出两匹马 X 和 Y 进行分析， X 和 Y 谁先谁后对后面的马儿没有影响。

当先放 X 时，Y 吃玉米的量为 $ta * ab$

当先放 Y 时，X 吃玉米的量为 $tb * aa$

当 $ta * ab < tb * aa$ 时，选择先牵 X ，反之，牵 Y 。

按照 $ta * ab < tb * aa$ 对结构体数组进行排序，然后贪心求解。

注意：最后结果可能超过int，所以建议定义ans为long long int类型！

B. Wiki with Card Game

算法：期望DP

数学期望 P = $\sum$ 每一种状态 $*$ 对应的概率。

比较裸的一道 dp 求期望次数，从前往后，或者从后往前推都可以(一般从后往前)：

(1) 从后往前，假设 $dp[i]$ 表示已经抽取i种不同名片，还需抽取的期望次数，$dp[n] = 0$;

$dp[i]$ = 第i个人取到已经取到过的名片 + 第i个人取到没有取到过的名片 $= i/n * dp[i] + （n - i）/n * dp[i + 1] + 1$。

每次抽取都百分百的增加抽取次数1。

化简得：$dp[i] = dp[i + 1] + n/(n - i)$

最后答案为 $dp[0]$。

（2)从前往后，假设 $dp[i]$ 表示抽出 i 种不同名片需要的期望次数，那么 $dp[1] = 1$;

$dp[i]$ = 前面$i - 1$个人取到各自不同的名片 + 第i个人取到第i种不同的名片 $=（i - 1）/n * dp[i - 1] +（n - i + 1）/n * dp[i] + 1$
。

化简得：$dp[i] = dp[i - 1] + n/(i - 1)$

最后答案即为 $dp[n]$。

C. Wiki with A|||B

考虑串 B 中每一位对答案的贡献单独计算。

为了方便计算，我们把两个串 $reverse$ 了。

如果 B 的第 k 位是 1 ，那么它对答案的贡献就是 $2^0 + 2^1 + \cdots + 2^k = 2^{(k+1)} - 1$。

如果 B 的第 k 位是 0 ，那么它对答案的贡献取决于串 A 的 $[0 \cdots k]$ 中的 1 的分布，为 $\sum_{i=0}^{k}[A_i == 1] \times 2^i$ 。

如果 B 的长度大于 A 的长度，那么多出来的几位的地位其实是和 B 的第 $|A|-1$ 位是一样的，把 k 看作 $|A|-1$ 算就行了。

D. Wiki with typhoon

龙卷风摧毁题解！

E. Yukina with blind box

如果现在已经有了 x 种款式，那么凑到 $x+1$ 种款式平均还需要 $n/(n-x)$ 瓶。

所以从没有款式开始，直到凑齐，共需要 $n*(1/1+1/2+\cdots+1/n)$ ——求调和级数

F. Yukina with Rhizomys

任务1：贪心，没到 k 种就继续放，到了就再开一个空间

任务2：dp，设 $dp[i]$ 表示前 i 只竹鼠的总危险值的最小值，显然 $dp[1]=1$

两层循环：

```
for(i=1...n)
    for(j=i...1)
        满足k的限制:
            dp[i]=min(dp[i],dp[j-1]+j到i作为一个空间时的危险度)
```

将第二层循环变量 j 看作当前考虑的空间(即将编号为 j 到编号为 i 的竹鼠看作放入一个空间)的起始位置

G1. Ranka with Matrix (easy version)

设 $sum[i][j][k]$ 表示以 $(1,1)$ 为左上角，(i,j) 为右下角的子矩阵中，所有大于等于 k 的数的和。

同样地，设 $num[i][j][k]$ 表示以 $(1,1)$ 为左上角，(i,j) 为右下角的子矩阵中，所有大于等于 k 的数的个数。

对于每个询问，最优策略一定是把子矩阵内的元素从大往小取，那么二分取的元素中最小的是多少即可。由于我们预处理了前缀和，所以我们能 $O(1)$ 判定。

总时间复杂度 $O(R\times C\times 1000+Q\times log_2 1000)$ 。

G2. Ranka with Matrix (medium version)

原题大放送 https://www.luogu.org/problem/P2468

主席树模板 https://www.luogu.org/problem/P3834

问题从矩阵退化成了一个序列，由于序列可能很长，普通的前缀和的时空复杂度已经无法忍受了。

我们需要一个在区间上能求前 k 大的和或个数的数据结构来。我们为序列开 C 棵在权值上的线段树。其中第 i 棵线段树存储序列 $[1,i]$ 上的值域的情况，线段树区间 $[l,r]$ 维护的信息为序列 $[1,i]$ 中，大小在 $[l,r]$ 内的元素的个数以及总和。这样做的好处是，我们维护的信息在区间上具有了可加性，也就是说，如果我们想要得到序列 $[L,R]$ 中，大于等于 x 的元素的总和，我们只需要求第 R 棵线段树上区间 $[x,1000]$ 的和减去第 $L-1$ 棵线段树上区间 $[x,1000]$ 的和即可。

总所周知，值域线段树的空间复杂度是 $O(\text{值域})$ 的，那么 C 棵就是 $O(C\times\text{值域})$ 的，似乎跟普通的前缀和比起来没有任何地长进？

我们注意到，第 i 棵线段树和第 $i+1$ 棵线段树很接近，因为后者只比前者多存了一个 $a[i]$，也就是说线段树上所有包含 $a[i]$ 的区间都不同，而其他区间都相同，而不同的区间只有 $O(log_{\text{值域}})$ 个，我们是不是只要新开 $O(log_{\text{值域}})$ 个结点就可以把整棵树存下来了？

于是我们得到了一个空间复杂度为 $O(C\times log_2 1000)$ 的，满足我们需求的数据结构。具体实现可以参考上面的模板题，网上也有很多代码。

剩下的做法完全一样，二分后可以利用主席树 $O(log_2 1000)$ $check$。总复杂度 $O(C\times log_2 1000+Q\times log_2^2 1000)$。

G3. Ranka with Matrix (hard version)

纪念
天国的 G3

如果加上修改操作呢？考虑到主席树本质上是一个前缀和，那么带修前缀和我们就自然想到了树状数组，我们给每个树状数组开一个动态开点的权值线段树，就得到了一个好用的树套树。虽然我写⑧完了，但是大家仍然有劲题可以做（狂喜）。

树套树模板题（动态区间 K 小) https://www.luogu.org/problem/P2617

H1. Ranka with Substring (easy version)

做法是枚举子串的出现位置更新答案即可。（当然找第一次出现和最后一次出现一定是最优的）

本来想整个 KMP ~~烤馍片~~，刻意卡了暴力，但是还是被怪怪的 $find$ 给过穿了。

H2. Ranka with Substring (medium version 1)

原题大放送 https://codeforces.com/contest/1203/problem/D2

我的赛时做法是，令 $pre[i]$ 表示 s 中第一个能够匹配 $t[0\cdots i]$ 的下标，$suf[i]$ 表示 s 中第一个能够匹配 $t[i\cdots len_t-1]$ 的下标。

这两个数组暴力 $O(len_s+len_t)$ 匹配就行了。

然后那么答案要么是 $pre[len_t-1]$ 的后面，要么是 $suf[0]$ 的前面，要么是 $pre[i]$ 到 $suf[i+1]$ 中间这一段，枚举取最小值即可。

有其他更为简短的做法，可以参考原题的 $editorial$。

H3. Ranka with Substring (medium version 2)

中午下楼吃了个麦当劳，结果这题就被各种爆破了。

这题需要一些前置芝士，如果理解了 $trie$ 和 KMP，那 AC 自动机就比较好理解了。

模板可以参考：

https://www.luogu.org/problem/P3808

https://www.luogu.org/problem/P3796

有一个问题就是，重复询问如果极多，且出现次数也极多，如果你在 $trie$ 的节点上开 $vector$ 暴力存所有以该点结尾的串的序号，就算是 AC 自动机也会 T 飞。只需要另开一个数组记一下它与谁是重复的即可。