分糖果

分类讨论题。计1、2和3颗的包数分别为c_1, c_2, c_3。

当$c_1 + c_2 \times 2 + c_3 \times 3$ 为奇数时，肯定不能平分为两部分。否则，按照c_1的取值进行分类讨论。

当$c_1 \geq 2$时，一定可以平分为两部分。考虑一种简单的贪心策略，先分配2和3颗一包的糖果，且每次都分配给糖果较少的一方，显然分配后两者的糖果数量之差的绝对值小于等于3。如绝对值之差为3，由于已经保证了总糖果数量为偶数，所以一定还有至少3包一个糖果；如绝对值之差小于等于2，一定还有至少2包一个糖果，此时仍然依次给较少的一方即可平分。

当$c_1 = 1$时，如果此时$c_2 \geq 1$，则一定可以平分，否则一定不可平分。执行之前的贪心策略，先分配3颗一包的糖果，再分配2颗一包的糖果。如果此时$c_2 \geq 1$，最后两者的糖果数量之差的绝对值一定为1；如果此时$c_2 = 0$，最后两者的糖果数量之差的绝对值一定为3。对于前一种情况可以平分，后一种情况不能平分。

当$c_1 = 0$时，此时只有2和3颗一包的糖果，如果两者的包数都有偶数个的话，显然可以平分；如果$c_2 \geq 3$且$c_3 \geq 2$的话，执行之前的贪心策略，可以平分。否则不能平分。

表达式求导

考虑导数的定义

由于本题只要求保留两位小数，将Δx带入为一个极小值后计算$f(x_0 + \Delta x)$和$f(x_0)$即可。于是问题从表达式求导转化为了表达式求值。

(好像出题人式子给的太简单了，导致赛中有些猛人过题是真的求出了求导后的式子，如果再给复杂点可能就会去想做表达式求值了)

置换操作

分类讨论题。合并相邻且相同的1，然后分为六种情况进行讨论。六种情况下得到的全1子串数量最大值即为答案。

第一种情况，需要特判下0的个数为0或1或n的情况；

第二种情况，在最长1串边上进行两次操作；

第三种情况，替换两串1之间的正好两个0；

第四种情况，替换两串1之间的正好一个0一次，在最长1串边上补一次操作；

第五种情况，替换两串1之间的正好一个0两次，且将3个1串组合并成1个1串；

第六种情况，替换两串1之间的正好一个0两次，且将4个1串组合并成2个1串。

游戏扑克牌

设f_i为黑色牌的数量正好为i个的方案数，因为已知黑色牌的数量至少为k_b，红色牌的数量至少为k_r，则黑色牌最终的数量只可能属于$[k_b, n-k_r]$。则只要求出来f_{k_b} f_{n-k_r}，答案即为最小且最靠前的$[L,R]$满足$\frac{\sum_L^R f_i}{\sum_1^n f_i} \geq \frac{p}{100}$。

由于其中有$0,1,\ldots,n$ 张黑色牌的概率相等，均为$\frac{1}{n+1}$，则可以直接通过$f_i = C(i,k_b)*C(n-i,n-k_r)$ 计算。

公切线

答案实际上只有3种情况，分别是4，3和Infinity。

先考虑Infinity如何判断，实际上只有两个正方形有且只有一个顶点重合时，才有无限条切线；

否则，如果一个正方形的一个顶点在另一个正方形的边上的话，则只有一条沿着边的内公切线和两条外公切线，共三条公切线；

再否则，如果两个正方形的各自某条边同时处于同一条直线上，则只有一条沿着直线的内公切线和两条外公切线，共三条公切线；

再否则，对于更一般的情况，两个正方形一定有两条内功切线和两条外公切线，共四条公切线。

Factorial

考虑对于每种素数直接算出答案取max，将数字看成p进制数，从高到低贪心。例如对于5，从1开始每125个数能提供31个5，之后的每25个数能提供6个5，再之后每5个数提供1个5。这样对于每个素数直接算出答案的话不会超时。

希望采用更直接的二分的话，需要一些剪枝卡常的技巧：例如按$p*e$排序，从大到小处理。二分的边界设为[当前答案,$min(1e18,p*e)$],类似的技巧卡一卡二分的边界，也可以通过。

(二分的复杂度是一个很满的log，想多一个log还能过题的话还是要有一定卡常的水平，不要写的太粗糙)

三元组

最优的构造一定是若干个1和若干个2，假设用a个2，答案是$a*(X-2*a)*(X-2*a-1)/2$ ，二分或者三分出极值点即可。也可以直接求导或者离线后利用决策的单调性优化掉这个log

(本来想将T加强要求必须均摊$O(1)$计算出来，但最后还是放了log过去)

二十四点

动态规划。比如$k=1$，设$f[i]$表示拼出来i所需最小1的数量，则初始状态为：$f[1]=1, f[11]=2, f[111]=3, f[1111]=4$

然后分别使用乘法和加法进行动态规划：

$$f[i] = min(f[i], f[j]+f[i-j]),\ \ 1<j<i$$

$f[i] = min(f[i], f[i/j] + f[j]),\ i$

暴力dp之，复杂度为$O(tn^2)$。但是由于$1 \le k \le 9$，因此预处理一下$f[k][i]$即可，其中$f[k][i]$表示对于$1 \sim 9$的k，拼出来i所需的最小k的数量。这样复杂度为$O(10n^2 + t)$。本地预处理了$0.5s$，如果只枚举i为k的倍数，就只需要跑$0.1s$。

(这个简单dp没人带榜做，大伙都在写分类讨论/卡常/模拟。)

字符串大师

如果t串两个位置字符相同，设为c。则应该把s串所有位置都改成这个字符c，第一问答案是$\frac{n*(n-1)}{2}$，第二问的答案是s串里不是c的位置数量。

如果t串两个位置字符不同，设为a和b，那么第一问答案是$\lfloor n/2 \rfloor * \lceil n/2 \rceil$，应该把左半部分改成$a$，右半部分改成$b$，如果$n$是奇数，中间位置可以是$a$或者$b$，扫一遍数一数即为第二问的答案。

kx+b数列

先考虑最一般的情况，列两个式子：

$k * A_1 + b = A_2$

$k * A_2 + b = A_3$

于是可以得到$k = (A_3 - A_2)/(A_2 - A_1), b = A_2 - k * A_1$ 。

也就是说，如果$A_3 \ne A_2$且$A_2 \ne A_1$且$(A_3 - A_2)$，则根据A_1, A_2, A_3可以得到一组合法的k和b，判断一下$A_4 \sim A_n$是不是也满足即可。

在此基础上，考虑一些特殊情况：

如果$n == 1$，因为只有一个数字，有多解，根据题目要求应该选1 0。

否则如果$a[2] == a[1]$，有多解，根据题目要求应该选1 0。

否则如果$n == 2$：

如果$a[1] == 0$，有多解，根据题目要求应该输出$1\ a[2]$。

否则如果$a[2] == 0$，有多解，应该输出$-1\ a[1]$或者$1\ -a[1]$，这和$a[1]$的正负有关。

否则，有多解，应该去找不定方程$k \times a[1] + b = a[2]$ 的$k\ b$的多解中符合题目要求的一个解。

（以下$n \ge 3$）

否则如果$a[3] == a[2]$，则没有k的非零解，输出-1。

否则如果$(A_3 - A_2)$，则没有k的整数解，输出-1。

否则，输出$k = (A_3 - A_2)/(A_2 - A_1), b = A_2 - k * A_1$ 。

需要在输出前判断一下是否$A_4 \sim A_n$也满足。

(验题人和选手都问了为什么没spj，有spj它就不是分讨题了。)

或的最大值

如果双重循环暴力，复杂度为$O(n^2)$，会超时。

接下来介绍两种做法：

第一种做法：有一个性质为：如果cnt个数字相加和为sum，则数字的不同种类数最多为$O(\sqrt{sum})$。于是使用$sort$，map等方法将数组去重，数组大小会减小为$O(\sqrt{\sum a_i})$约为10^4，此时双重循环即可通过。

第二种做法：根号分治。考虑令$m = 10^4$，将小于m的数字用桶存一下，大于m的数字存在数组里。则小于m的数字缩减到了10^4个；由于$\sum a_i \le 10^8$，因此大于m的数字数量也为$O(\frac{10^8}{m})$也是10^4个，此时双重循环即可通过。

（开赛后发现了有若干错误做法的问题，赛中构造了一些hack数据，由于本题是签到，提交量比较大，提交比较密集，为了避免重测导致服务器波动影响其他题目，赛中没做处理。）

士兵列队

为了满足要求，需要n的二进制表示下1的数量小于等于4且$n \ge 4$。如果不满足，往n上加$lowbit(n)$即可。

由于每次$+lowbit()$后，二进制表示下最低位的1单调增大，因此复杂度为$O(log_2(n))$。

集合游戏

重新定义一下游戏操作：不断从所有非空集合中等概率选择一个集合，然后删除一个该集合中的元素。

然后我们生成一个长度为 $n+m$ 的操作序列，当操作序列最后一次不是删除 T 集合中元素时，我们称该操作序列是合法的，那么本题求的就是所有合法操作序列在原题中生成的概率之和。

考虑枚举一个操作序列删完 S 后 T 集合的大小，假设为 i，那么该序列在原题中被生成的概率就是 $\dfrac{1}{2^{n+m-i}}$。而这样的序列有 $\dbinom{n+m-i-1}{n-1}$ 个，这是由于第 $n+m-i$ 次操作必须删 S 中的元素，前 $n+m-i$ 次操作删了 n 次 S 中的元素。那么答案就是：

$\sum_{i=1}^{m}\dfrac{1}{2^{n+m-i}}\binom{n+m-i-1}{n-1}$ $=\dfrac{1}{2^{n+m}}\left(\sum_{i=0}^{m}2^i\binom{n+m-i-1}{n-1}-\binom{n+m-1}{n-1}\right)$ $=$ $\dfrac{1}{2^{n+m}}\left(\sum_{i=0}^{m}\sum_{j=0}^{i}\binom{i}{j}\binom{n+m-i-1}{n-1}-\binom{n+m-1}{n-1}\right)$ $=\dfrac{1}{2^{n+m}}\left(\sum_{i=0}^{m}\sum_{j=i}^{m}\binom{j}{i}\binom{n+m-j-1}{n-1}-\binom{n+m-1}{n-1}\right)$ $=$ $\dfrac{1}{2^{n+m}}\left(\sum_{i=0}^{m}\binom{n+m}{i+n}-\binom{n+m-1}{n-1}\right)$ $=\dfrac{1}{2^{n+m}}\left(\sum_{i=0}^{m}\binom{n+m}{i}-\binom{n+m-1}{n-1}\right)$

其中倒数第二步由范德蒙德卷积可得。

然后发现答案是一行组合数的前缀和与一堆系数。可以莫队优化，复杂度 $\mathcal{O}(n\sqrt{n})$。