t1

我们有结论：

$$\left\lfloor \frac{n}{xy} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{n}{x} \right\rfloor}{y} \right\rfloor$$

所以不难发现优惠券不需要一次结算，每一次结算除 2 即可。然后就变成了维护集合，每次找最大值除 2 的问题。用大根堆轻松实现。

t2

考虑枚举每个点，计算有多少子集包括它。这个点向上向下向左向右将坐标轴分成了四部分，如果我们能快速求出这四部分中点的个数，即可通过分类讨论算出哪些是不合法的，即只在一个区域内，或者在相邻的两个区域内。排除掉这些子集即可。

接下来问题就变为了计算四个区域内的点个数，这是经典的二维数点，排序一维后另一位用树状数组维护即可。

t3

先说结论，如果把 `A` 看作 1，`B` 看作 2，则两个字符串可以相互转化当且仅当它们所有字符的和模 3 相同。

由操作，必要性显然。下证充分性。

首先可以构造出操作是可逆的，所以可以先全部变为 `A`，然后在这个意义下比较（反正可以逆回去），显然模 3 相同即可。充分性得证。

t4

类似数位 dp，我们枚举比 n 小的位所在位置 p，和众数 m 以及其出现次数 c。对于剩下需要填的 $p+1$ 到 $|n|$ 位，我们直接考虑多重背包，依次考虑除了众数外的每个数码：

$$f_{i,j} = \sum_k f_{i-1,j-k} \times \binom{j}{k}$$

根据 p 之前出现过的数码个数和 c，我们不难求出每个 k 的上限。

最终答案即：

$$f_{9,(n-p+1)-(c-c')} \times \binom{n-p+1}{c-c'}$$

其中 c' 为前 $p-1$ 位众数出现次数。

上述做法求出了与 n 长度相同的数。如果长度不同，枚举长度和第一位填什么即可规约到上述情况。

每次背包的复杂度是 $\mathcal{O}(d|n|^2)$，算上枚举长度（或者 p），枚举众数和其出现次数，则总复杂度为 $\mathcal{O}(d^3|n|^4)$。

但是相信数位 dp 的小常数，我们还可以在不改变复杂度的情况下减少一些计算次数。对于 $\binom{j}{k}$ 中的 $\frac{j!}{(j-k)!}$，不难验证，在背包的限制下，它最终会乘出一个 $((n-p+1)-(c-c'))!$，相当于分段统计这个阶乘。所以最终乘一次即可，大大减小运算次数。可以通过。

t5

不难写出一个 dp，设 $f_{i,j}$ 表示处理了前 i 个积木，且第 i 个积木左边界在 j，则有转移：

$$f_{i,j} = |j-l_i| + \min_{j-len_{i-1} \le k \le j+len_i} \{f_{i-1,k}\}$$

可以发现 $f_i(j)$ 是凸函数。因为绝对值 $|j-l_i|$ 是凸的，递归证明后面一项相当于对凸函数做了偏移，仍然是凸函数，二者相加也是凸的。（递归边界不再赘述）

然后我们从函数角度考察上述转移，如果有 $f_{i-1}(j)$ 的最小值为 v 且所在区间为 $[L,R]$，则转移形如：

$$f_i(j) = |j-l_i| + \begin{cases} f_{i-1}(j+len_i) & j < L-len_i \\ v & L-len_i \le j \le R+len_{i-1} \\ f_{i-1}(j-len_{i-1}) & j > R+len_{i-1} \end{cases}$$

即根据上述 k 能否取到最小值分类。因为是凸函数，所以越靠近最小值越好。

如果不考虑绝对值，实际上就是把最小值铺的更开了一点，即 L 变为 $L-len_i$，R 变为 $R+len_{i-1}$，其余相应延伸。

处理好 L,R 的变化后，考虑 $|x-l_i|$。实际上就是把 l_i 两边的函数斜率 -1 或 $+1$。考虑 l_i 的位置：

- 如果在斜率为 0 的地方，即最小值，则此时最小值只剩它一个了，左边 -1，右边 $+1$ 即可。
- 如果在 L 左边，则会出现一段斜率为 -1 的变为 0。设这段 -1 为 $[s,t]$，则新的 L,R 分别为 $\max(s,l_i),t$。然后处理好剩下的 $+1/-1$ 即可。
- 在 R 右边同理。

上述过程可以用堆维护。维护好 L 左边和 R 右边的拐点，找 -1 区间实际上就是前两个拐点。整体偏移维护标记即可，