# Problem Tutorial: “Чаювання в Хмеляндії”

Оскільки Хмелюша вип'є весь чай, який буде налитий до склянки, то можемо рахувати для кожного сорту чаю кількість цього чаю, яка буде налита до склянки. Нехай наша склянка має місткість два. Початково склянка буде наповненка чаєм номер 1 (тобто +2 до чаю номер 1). Доливання чаю в якомусь кроці додає +1 до кількість налитого чаю певного сорту. Потім просто вибираємо той чай, якого є найбільше (якщо таких сортів кілька - то той, номер якого найменший).

Дане рішення легко написати, якщо користатися зі структури map в C++.

# Problem Tutorial: “Мурахи на колі”

Ключовим спостереженням до цього завдання буде наступне: можна вважати, що мурашки незалежні і не відбиваються один від одного. Якщо в якомусь моменті дві мурашки зустрічаються, то можна вважати, що вони заміняються своїми номерами і йдуть далі своїм шляхом без жодного відбиття (все одно треба буде позиції посортувати).

Після цього можемо легко в складності $O(1)$ визначити позиціїї мурашок через $T$ секунд танцю (якщо пронумеруємо позиції від 0 до $N-1$ - то відповіддю буде $(startposition + T * direction) mod N$.

Далі залишається просто посортувати ці значення.

# Problem Tutorial: “Хмелюша і матриці”

Рішення за $O(N^2 * M)$:
Можна було зафіксувати два рядки (назвемо їх $i \leq j$).
Тоді хочемо відповісти на питання: скільки є добрих прямокутників таких, що верхній відрізок прямокутника лежить на $i$, нижній - на $j$. Тоді нехай $d_x = (\sum_{r=i}^{j} A[r][x]) mod 2$. Більше того, нехай $d_0 = 0$.
Нехай $c_1 = \sum_{r=0}^{M} d_x, c_0 = M + 1 - c_1$. Тоді шукана кількість буде $c_0 * (c_0 - 1)/2 + c_1 * (c_1 - 1)/2$ (якщо більш інтуїційно - то рахуємо масив часткових сум по модулю два).

Рішення за таку саму складність, але швидше:
легко бачити, що переходячи від рядка до наступного це все можна тримати на бітсетах. Тоді складність буде десь в районі $O(N^2 * M/32) = O(N^2 * M)$. Але стала буде дуже малою, що давало максимальну кількість балів в цій задачі.

# Problem Tutorial: “Прямолінійність в Хмеляндії”

<trololo solution mod on>
Якщо піднесемо наступну матрицю до відповідної степені, то можемо легко відновити відповідь.

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 2 & 2 & 2 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 2 & 2 & 2 & 1 & 1 & 1
\end{pmatrix}$$

</trololo solution mod on>
Нормальне рішення:
потрібно було зауважити подібність даної послідовності до чисел Фібоначчі. Розглянемо легшу задачі: знайти $n$-те число Фібоначчі швидко (за логарифм, скажемо). Можна зауважити наступну залежність:

$$\begin{bmatrix} f_{n+1} \\ f_n \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} f_n \\ f_{n-1} \end{bmatrix}$$

Маючи таку залежність можна просто обрахувати відповідне число Фібоначчі, бінарно підносячи матрицю до степені. Тобто маючи такі лінійні залежності ми можемо швидко рахувати щось.

Але наше завдання є трошки складнішим, тому що тут треба рахувати суму, ще й квадратів.
Нехай $S_N = T_1^2 + T_2^2 + ... + T_N^2$. Тоді ми маємо щось таке:
$S_N = S_{N-1} + T_N^2$.
$T_N^2 = (T_{N-1} + T_{N-2} + T_{N-3})^2 =$
$= T_{N-1}^2 + T_{N-2}^2 + T_{N-3}^2 + 2 * T_{N-1} * T_{N-2} + 2 * T_{N-1} * T_{N-3} + 2 * T_{N-2} * T_{N-3}$.
Тобто $S_N$ залежить від $S_{N-1}$ і $T_N$.
А $T_N^2$ залежить від $T_{N-1}^2, T_{N-2}^2, T_{N-3}^2, T_{N-1}*T_{N-2}, T_{N-1}*T_{N-3}, T_{N-2}*T_{N-3}$. Подібно треба розписати $T_N * TN - 1, T_N * T_{N-2}$ і інше.
Тоді можна це все обчислювати за допомогою піднесення матриці до степеня.

# Problem Tutorial: "Гопнік Вася"

Як можна було думати під час розв'язування цієї задачі: нехай Вася крутий і дав нам такий blackbox - функцію $F(r)$, яка каже нам кількість чисел від 1 до r, які не викличуть у Васі агресію (закладаємо, що ця функція працює швидко). Зауважимо, що така функція є монотонічною. Тоді можна просто застосувати бінарний пошук і знайти відповідь.

Як написати таку функцію: можна застосувати програмування динамічне. Будемо будувати число цифра по цифрі, починаючи від найбільш значимих. Нехай $f[n][last][strict][nonzero]$ - кількість таких чисел, що ми вже збудувати префікс довжини $n$, остання цифра була $last$, $strict$ каже нам чи те, що ми вже збудували є префіксом числа $r$, nonzero - чи була ненульова цифра. Тоді для кожного стану можемо перебрати цифру, яку будемо ставити на теперішню позицію, і робитимемо перехід, якщо можемо (тобто, пара $last$ і $digit$ не викличе у Васі агресію + те число, яке ми будуємо, не стане більше за $r$). Переходи динаміки залишаємо читачеві як просту вправу.

Тоді функція Васі працює за складність $O(log(r))$, і наше рішення є швидким =)