Algorithmic Leo Camp

День 2

Розбір задач

H - Шахівниця

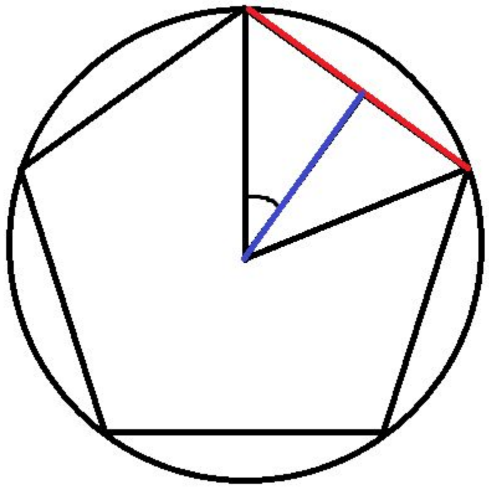
- Занумеруємо букви a, b, c, …, h числами 1, 2, 3, …, 8.
- Тепер координати кожного поля задаються двома числами x, y.
- Чорні поля мають парну суму $x + y$, а білі - непарну.

E - Велосипед

- На передню зірку ми можемо використати всі a варіантів, а на задню b.
- Використаємо правило добутку.
- Відповіддю буде $a \cdot b$.

A - Задача Архімеда

- При збільшенні *n* площа прямує до π.
- Відповідь рівна
  $(\cos(pi / n) \cdot (2 \cdot \sin(pi / n)))/2 \cdot n$
  $= \sin(2 \cdot pi/n) \cdot n/2$



G - Дивні операції

- Розгляньмо числа у двійковій системі числення.
- Кожне число має не більше ніж 30 бітів.
- На *i*-тому кроці будемо занулювати *i*-тий біт, вибравши $d = 2^{(30-i)}$.
- Якщо цей біт у числа - 1, то робимо -*d*, якщо 0 - то нічого.

K - Система рівнянь

- Якщо a і b різної парності, то відповіді не існує.
- Якщо $a < b$, то відповіді не існує.
- Інакше виведемо b, $(a-b)/2$, $(a-b)/2$.

L - Система рівнянь 2

- З попередньої задачі відомо, що якщо a і b мають різну парність або $a < b$, то розв’язків немає.
- Нехай $f(a, b)$ - відповідь на задачу.
- Базовий випадок $f(0, 0) = 1$, бо існує єдиний розв’язок $x = y = z = 0$.

L - Система рівнянь 2

- Якщо числа *a* і *b* парні, то серед чисел *x, y, z* немає жодного непарного або рівно 2 непарні.
- Якщо *x, y, z* парні, то $f(a, b)=f(a/2, b/2)$
- Якщо ж серед *x, y, z* є 2 непарних числа, то є три способи вибрати, які саме числа будуть непарні. $f(a, b)=3\cdot f((a-2)/2, b/2)$

L - Система рівнянь 2

- Нам потрібно підтримувати однакову парність a і b, тому можемо однозначно визначити, скільки чисел x, y, z будуть непарними і перейти до відповідної задачі.

L - Система рівнянь 2

- Аналогічно розглядається випадок, коли a і b непарні.
- Якщо серед x, y, z рівно 1 непарне число, то вибираємо трьома способами, яке це число. $f(a, b)=3 \cdot f((a-1)/2, (b-1)/2)$
- Якщо всі числа x, y, z непарні, то $f(a, b)=f((a-3)/2, (b-1)/2)$
- Знову лише один випадок збереже однакову парність нових параметрів.

I – Фірмові браслети

- Застосуємо жадібну стратегію.
- Кожний підрядок ALC в початковому рядку додає 1 до відповіді.
- Жадібно з'єднаємо в браслет пари рядків що закінчуються на AL та починаються на C, а також пари що закінчуються на A і починаються на LC.
- З'єднаємо трійки невикористаних рядків x, y, z, такі що x закінчується на A, y = “L”, z починається з C.

J - Щасливий Данило

- Використаємо підхід динамічного програмування.
- $dp[i][j][flag]$ - максимальна відповідь, якщо йти з клітинки i, j до виходу.
- $flag$ позначає, куди ми робитимемо наступний крок (нехай 0 - вправо, 1 - вниз).
- $dp[i][j][0] = max(dp[i][L][1] + sum(j, L))$ де $sum(j, L)$ - це сума елементів від j до L, і $j < L \leq j+k$, перехід по прапорцю 1 аналогічний.

J - Щасливий Данило

- Нам потрібна структура даних яка вміє швидко шукати максимум на відрізку.
- Можна використати дерево Фенвіка, дерево відрізків, або чергу на двох стеках.
- Порахуємо префіксні суми по кожному рядку й стовпцю.
- Переформулюємо перехід.
- *dp[i][j][0] = max(dp[i][L][1] + pref[i][L]) - pref[i][j]*, перехід по 1 аналогічний.

J - Щасливий Данило

- Загальна складність $O(n \cdot m)$ із чергою, або $O(n \cdot m \cdot (\log(n)+\log(m)))$ з деревом Фенвіка або деревом відрізків.

F - Кількість розв'язків рівняння

- Застосуємо динамічне програмування.
- $dp_i[sum]$ - скільки є способів набрати суму sum, розставивши перші i змінних.
- $dp_i[0]=dp_{i-1}[0]$
- $dp_i[1]=dp_{i-1}[0]+dp_{i-1}[1]$
- ...
- $dp_i[c_i]=dp_{i-1}[0]+dp_{i-1}[1]+\ldots+dp_{i-1}[c_i]$
- $dp_i[c_i+1]=dp_{i-1}[1]+dp_{i-1}[2]+\ldots+dp_{i-1}[c_i+1]$
- $dp_i[c_i+2]=dp_{i-1}[2]+dp_{i-1}[3]+\ldots+dp_{i-1}[c_i+2]$
- ...

F – Кількість розв'язків рівняння

- Зауважимо, що $dp_n[x]$ не залежить від порядку змінних.
- При заміні c_i на x порахуємо dp_{n-1} з dp_n вважаючи, що $c_n = c_i$. Потім порахуємо dp_n з dp_{n-1} вважаючи, що $c_n = x$.
- Загальна складність $O((n + q) \cdot k)$.

D - Марічка вмикає лампи

- Відповідь на задачу дорівнює $\sum_{k=0}^{n} \frac{C_n^k}{C_m^k}$
- $$\frac{C_n^k}{C_m^k} = \frac{n!(m-k)!}{(n-k)!m!} = \frac{\frac{(m-k)!}{(n-k)!(m-n)!}}{\frac{m!}{n!(m-n)!}} = \frac{C_{m-k}^{n-k}}{C_m^n}$$
- $$\sum_{k=0}^{n} \frac{C_{m-k}^{n-k}}{C_m^n} = \frac{1}{C_m^n} \sum_{k=0}^{n} C_{m-k}^{n-k}$$
- $$C_n^k = C_{n-1}^k + C_{n-1}^{k-1} = C_{n-1}^k + C_{n-2}^{k-1} + C_{n-2}^{k-2} = \cdots = C_{n-1}^k + C_{n-2}^{k-1} + \cdots + C_{n-k-1}^0$$
- $$\frac{1}{C_m^n} \sum_{k=0}^{n} C_{m-k}^{n-k} = \frac{C_{m+1}^n}{C_m^n} = \frac{m+1}{m-n+1}$$

B - Не згортка

- Порахуємо $f_i[d]$ для числа i та його дільника d - це сума елементів $b[j]$ таких, що $gcd(i, j) = d$.
- Нехай $e[d]$ - сума $b[j]$ таких, що j ділиться на d
- Початково зробимо $f_i[d] = e[d]$.
- Розглядаємо дільники числа i від більших до менших.
- Тоді по кожному дільнику d_1 числа d зробимо $f_i[d_1]$ -= $f_i[d]$.

C - Щасливі підпослідовності

- Нехай $dp_i[c]$ - кількість підпослідовностей, на префіксі i, що закінчуються на щасливий символ c.
- $dp_i[c] = dp_{i-1}[c]$, якщо $c \neq s_i$.
- $dp_i[s_i] = 1 + \Sigma_c\, dp_{i-1}[c]$.
- Вектор dp_i одержується з dp_{i-1} множенням на деяку матрицю A_i.
- Зауважимо, що до векторів dp_i треба дописати одиницю, щоб можна було множити матрицю.

C - Щасливі підпослідовності

- Для відповіді на запит треба помножити матриці на проміжку *[l, r]* на початковий вектор $(0, 0, 0, 0, 0, 1)^T$.
- $A_r \cdot A_{r-1} \cdot \ldots \cdot A_l \cdot v$ - кінцевий вектор, відповідь буде рівна сумі перших 5 значень вектора.
- $A_r \cdot A_{r-1} \cdot \ldots \cdot A_l \cdot v = A_r \cdot A_{r-1} \cdot \ldots \cdot A_l \cdot (A_{l-1} \cdot \ldots \cdot A_1 \cdot A_1^{-1} \cdot \ldots \cdot A_{l-1}^{-1}) \cdot v = (A_r \cdot A_{r-1} \cdot \ldots \cdot A_l \cdot A_{l-1} \cdot \ldots \cdot A_1) \cdot (A_1^{-1} \cdot \ldots \cdot A_{l-1}^{-1}) \cdot v = B_r \cdot C_{l-1} \cdot v$, де

  $B_i = A_i \cdot A_{i-1} \cdot \ldots \cdot A_1$, а $C_i = A_1^{-1} \cdot A_2^{-1} \cdot \ldots \cdot A_i^{-1}$