Maratón nacional
Association for Computing Machinery - Colombia

Maratón nacional - Association for Computing Machinery Colombia

Noviembre 23 de 2024

Soluciones

Fase final - Categoría basica

A. Juego de cartas (Easy version)

Autor: Wilmer Castrillon, Universidad de la Amazonia

Se tiene un numero ilimitado de cartas de Poker, cada carta pertenece a un palo (picas, corazones, diamantes o tréboles) y tiene un numero entre 1 y K, se quiere contar cuantas pilas ordenadas diferentes de tamaño N se pueden hacer, es decir de cuantas formas diferentes se pueden organizar N cartas de forma creciente, para el ordenamiento no importa el palo (ejemplo, 2 de corazones puede estar arriba o abajo del 2 de picas) pero para el conteo si se debe considerar (el 2 de corazones es diferente al 2 de picas).

Una primera observación es que el palo de cada carta no influye en el orden de las cartas, entonces se requiere contar cuantas pilas ordenadas pueden existir con números desde 1 hasta K (Sea C ese numero de combinaciones), y a cada pila se le pueden asignar todas las combinaciones de palos, para cada carta existen 4 opciones, entonces la respuesta seria $C * 4^N$ y faltaría calcular C.

Siguiente observación, sea A una pila ordenada, el numero A_i debe ser mayor o igual a A_{i-1}, A se puede partir en K intervalos, donde cada intervalo tendrá un mismo numero, por ejemplo si $N = 4$ y $K = 2$ entonces podemos fijar un punto j en el cual todos los anteriores sean 1 y los demás 2, si $j = 2$ entonces $A = 1, 2, 2, 2$ (el intervalo de 1 es $[1, 1]$ y el de 2 es $[2, 4]$), si $j = 3$ entonces $A = 1, 1, 2, 2$ (el intervalo de 1 es $[1, 2]$ y el de 2 es $[3, 4]$) y así sucesivamente, se puede observar que para $j = 1$ el intervalo de números 1 seria vacío, es decir no hay ninguna carta con numero 1, o cuando $j = 5$ se representa una pila sin cartas 2, ambos casos son pilas validas.

Como tenemos una pila A de tamaño N y necesitamos partirla en K intervalos, se convierte en un problema de combinatorias (teniendo n elementos contar de cuantas formas se puede escoger k elementos), podemos imaginar los $K-1$ puntos de corte como elementos en n (ademas de las N cartas), es decir teniendo $N+K-1$ elementos vamos a escoger $K-1$ elementos, entonces la respuesta final es calcular $\binom{N+K-1}{K-1} * 4^N$

Para el calculo de la respuesta se puede aplicar directamente la formula de combinatorias, como la formula incluye una división se debe utilizar el inverso modular.

B. Volumen ideal

Autor: Sneider Quintero, Universidad de la Amazonia

Note que una de las claves del problema es ver que los volúmenes que no se les realiza multa, comprenden un prefijo del volumen total.

Teniendo esto en cuenta, se puede hacer *Binary search* en la respuesta. Hay que tener cuidado con el movimiento de los indices, ya que puede llegar a caer en un volumen que no es el correcto, asegúrese de siempre asignar $l = mid$ o $r = mid$ en lugar de $mid \pm 1$.

C. Domir en el bus es malo

Autor: Sebastian Galindo, Universidad Javeriana

Primero que todo, fije la primera parada como el tiempo 0 y haciendo el prefix sum de los tiempos se obtiene el tiempo desde el origen hasta una parada. Denote que para cada query se tiene que encontrar un x tal que:

$$p_i + (k + 2k + ... + xk) = p_f$$

La ecuacion anterior se puede reordenar de la siguiente forma:

$$p_i + (k + 2k + ... + xk) = p_f \tag{1}$$

$$p_i + k(1 + 2 + ... + x) = p_f \tag{2}$$

$$p_i + k\left(\frac{x(x+1)}{2}\right) = p_f \tag{3}$$

Teniendo esta ecuación, se puede resolver para x, la cual sería la cantidad de veces el cual se durmió en el camino. Solucionando para x da la siguiente formula:

$$x = \frac{\sqrt{1 + \frac{8(p_f - p_i)}{k}} - 1}{2}$$

Si x no es entero, entonces la respuesta es `NO`. En el caso contrario la respuesta es `YES` y se imprime el valor dado de x. La complejidad para solucionar cada query debe ser de $O(1)$

D. Ni sabemos donde va a caer

Autor: Sebastian Galindo, Universidad Javeriana

Una observación y/o dato importante para resolver este problema es el siguiente:

La distancia mínima que tiene que recorrer cualquier punto P para llegar al circulo C se obtiene trazando una recta desde el origen que pase por el punto P. La intersección con el circulo C es el punto mas cercano del circulo a P.

Ya con esto, podemos buscar minimizar la siguiente expresión del circulo:

$$|x^2 + y^2 - R|$$

Puede interpretar esta expresión como que de alguna manera, La distancia desde el origen hasta el punto (x, y) se le resta la recta formada desde el origen hasta el punto mas cercano del circulo, lo cual es equivalente al radio $\sqrt{R}$.

Entonces, podemos buscar x y calcular y con respecto a la expresión anterior:

$$y = \sqrt{R - x^2}$$

Denote que los valores posibles de x están en el rango $[0, \sqrt{R}]$. Una nota muy importante es que en ningún momento en el enunciado se menciona que el punto debe estar en el circulo. Por lo tanto, hay que tomar los valores de $\lfloor y \rfloor$ y $\lceil y \rceil$, ya que los puntos con la aproximación hacía arriba pueden ser validos.

Complejidad: $O(\sqrt{R})$

E. Roll N Dash

Autor: Pedro Barrera, Universidad Distrital Francisco José de Caldas

Este problema solo requiere de algebra y transformaciones lineales, además de unos cuantos condicionales para simplificar casos que no requieren calculos.

Lo primero es determinar el movimiento lineal del objeto a desplazar. Como todo el objeto va a tener el mismo desplazamiento, entonces basta con calcular el recorrido lineal de un punto. Por simplicidad se toma el punto centro del objeto, que finaliza su recorrido en el origen del plano. Así, el recorrido es la recta $y = mx$ con pendiente

$$m_c = \frac{y_c}{x_c}$$

donde x_c y y_c son las componentes del punto centro del objeto a desplazar.

Ahora, tomando los puntos extremos del obstáculo $P_1 = (x_{p_1}, y_{p_1})$ y $P_2 = (x_{p_2}, y_{p_2})$ se pueden obtener las rectas l_1 y l_2 de la forma $y = mx + b$ con pendiente m_c que pasan por P_1 y P_2, respectivamente. Entre estas dos rectas se forma un area infinita no definida que vamos a llamar area de obstrucción o A_o (al ser no definida, $l_1 \notin A_o$ y $l_2 \notin A_o$).

Luego, tomando los puntos extremos del objeto a desplazar $P_3 = (x_{p_3}, y_{p_3})$ y $P_4 = (x_{p_4}, y_{p_4})$, y el punto centro del objeto $P_c = (x_c, y_c)$, se debe determinar el estado del sistema, el cual puede tener tres valores:

1. El obstáculo impide definitivamente el movimiento de dash: Este estado se da cuando

$$(P_c \in A_o) \wedge \left(\exists P_i \mid \left(P_i \in \overline{P_cO} \wedge P_i \in \overline{P_1P_2}\right)\right)$$

2. El obstáculo impide el dash directo, pero se puede rotar el objeto para quedar en una posición que posibilite el dash: Este estado se da con dos posibles condiciones. La más identificable es

$$\bigvee_{j=3}^{4} \left((P_j \in A_o) \wedge \left(\exists P_i \mid \left(P_i \in \overline{P_1P_2} \wedge \overline{P_cO} \parallel \overline{P_jP_i} \wedge \left|\overrightarrow{P_cO}\right| \leq \left|\overrightarrow{P_jP_i}\right|\right)\right)\right)$$

   Realmente esta condición se da con un o exclusivo, sin embargo, la condición del estado 1 impide que se llege a un escenario en que la condición se cumpla para $j = 3$ y $j = 4$ al tiempo. La segunda condición que origina este estado se expresa matematicamente como

$$\exists k \mid \left(\left(P_k \in \overline{P_3P_4} \wedge P_k \notin \{P_3, P_4\} \wedge P_k \in A_o\right) \wedge \left(\exists P_i \mid \left(P_i \in \overline{P_1P_2} \wedge \overline{P_cO} \parallel \overline{P_kP_i} \wedge \left|\overrightarrow{P_cO}\right| \leq \left|\overrightarrow{P_kP_i}\right|\right)\right)\right)$$

   Realmente en esta condición P_k debe ser diferente de P_c, sin embargo, la condición del estado 1 impide que se llege a este escenario. Las dos condiciones generan el mismo estado y se resuelven de la misma forma teórica, con una sutíl diferencia en la práctica en la que se ahondará adelante.

3. El obstáculo no impide realizar el dash: Este estado se puede presentar con distintas configuraciones del sistema, sin embargo con las dos condiciones anteriores es suficiente para confirmar si este estado se da por omisión. Como recomendación al implementar la solución se pueden identificar algunas configuraciones que llegan a este estado con la condición

$$k \mid \left(P_k \in \overline{P_3P_4} \wedge P_k \in A_o\right)$$

Ahora, habiendo identificado el estado del sistema, se pueden generar las respuestas. Para los estados 1 y 3 las salidas son evidentes. Para el estado 2 hay dos posibilidades. Si la condición que lo identificó es la primera, se requiere conocer el j que cumplió la condición. Si se cumple la segunda condición se debe identificar $j \in \{3, 4\} \mid P_k \in \overline{P_cP_j}$.

Con j identificada, la solución se reduce a identificar los ángulos ψ_o tales que permiten el dash. Para identificarlos se deben definir los puntos $P_s = (x_s, y_s)$ y $P_j = (x_j, y_j)$ para los cuales se cumple que

$$\begin{bmatrix} x_s \\ y_s \end{bmatrix} = \begin{bmatrix} cos(\psi_o) & -sen(\psi_o) \\ sen(\psi_o) & cos(\psi_o) \end{bmatrix} \begin{bmatrix} x_j - x_c \\ y_j - y_c \end{bmatrix} + \begin{bmatrix} x_c \\ y_c \end{bmatrix}$$

Operando se tiene

$$x_s = (x_j - x_c)cos(\psi_o) - (y_j - y_c)sen(\psi_o) + x_c$$

$$y_s = (x_j - x_c)sen(\psi_o) + (y_j - y_c)cos(\psi_o) + y_c$$

Despejando se tiene

$$cos(\psi_o) = \frac{y_s - y_c - (x_j - x_c)sen(\psi_o)}{y_j - y_c}$$

$$x_s = (x_j - x_c)\frac{y_s - y_c - (x_j - x_c)sen(\psi_o)}{y_j - y_c} - (y_j - y_c)sen(\psi_o) + x_c$$

$$x_s = \frac{(x_j - x_c)(y_s - y_c)}{y_j - y_c} - sen(\psi_o)\left(y_j - y_c + \frac{(x_j - x_c)^2}{y_j - y_c}\right) + x_c$$

$$\psi_o = sen^{-1}\left(-\frac{x_s - x_c - \frac{(x_j - x_c)(y_s - y_c)}{y_j - y_c}}{y_j - y_c + \frac{(x_j - x_c)^2}{y_j - y_c}}\right)$$

De esta ecuación los terminos desconocidos son x_s y y_s, que son las componentes del punto solución P_s. Este punto es aquel al que se puede rotar P_j que no esté en el area A_o. Para hallarlo se puede plantear la circunferencia c con centro en P_c y radio $\left|\overrightarrow{P_cP_j}\right|$ cuya ecuación es

$$(x - x_c)^2 + (y - y_c)^2 = \left|\overrightarrow{P_cP_j}\right|^2$$

Así P_s se definen como $P_s \in c \wedge (P_s \in l_1 \vee P_s \in l_2)$.

Para mostrar cómo hallar las componentes de P_s se va a tomar una recta arbitraria l con la forma $y = mx + b$ con pendiente m_c.

$$y = m_c x + b$$

$$(x - x_c)^2 + (m_c x + b - y_c)^2 = \left|\overrightarrow{P_cP_j}\right|^2$$

$$x^2 - 2xx_c + x_c^2 + m_c x^2 + 2m_c bx + b^2 - 2y_c mx - 2y_c b + y_c^2 = \left|\overrightarrow{P_cP_j}\right|^2$$

$$\left(1 + m_c^2\right)x^2 + \left(-2x_c + 2m_c b - 2y_c m_c\right)x + \left(x_c^2 + b^2 - 2y_c b + y_c^2 - \left|\overrightarrow{P_cP_j}\right|^2\right) = 0$$

La ecuación es una cuadrática, donde b debe ser probado para el término b usado en l_1 y para l_2. Luego, hallar la componente y es casi trivial.

Por último, debido a que $\psi \in Z$, pero debe ser el menor de los ángulo optativos ψ_o, debe evaluarse si aplicando función mayor entero o menor entero se cumplen las condiciones para determinar el valor de ψ.

F. Siempre hay un sapo (Easy version)

Autor: Sebastian Galindo, Universidad Javeriana

Note que para esta versión, no hay que pagarle nada a los sapos, entonces lo mejor es tomar una estrategia greedy.

Puede ir verificando para cada uno de los estudiantes su valor a_i, si $a_i \geq b_i$ entonces siempre va a ser mejor tomar a_i, osea, que el estudiante se quede en el salón de Sebastian, en el caso contrario ($b_i > a_i$), entonces es mejor tomar b_i, quiere decir, mandar el estudiante para el otro curso.

Complejidad: $O(n)$.

G. Consulta de letras

Autor: Sneider Quintero, Universidad de la Amazonia

Note que, para cada una de las preguntas, no es viable verificar el rango linealmente, ya que la complejidad de la solución quedaría $O(nq)$, lo cual sobrepasa el tiempo limite del ejercicio.

Para cada una de las letras, puede precalcular sus ocurrencias usando un *prefix sum*. Eso quiere decir, que puede mantener 26 vectores de tamaño $|S|$ donde almacene el prefix sum de cada una de las letras. Por ejemplo, si esta evaluando la letra `a`, puede recorrer linealmente el string S como si fuera un arreglo de $1's$ y $0's$, donde el 1 significa que la letra aparece en la posición que se esta evaluando del string. Note que si usted realiza el prefix sum del arreglo anteriormente descrito, con una range query ya puede saber cuantas ocurrencias de la letra hay en el rango especificado.

Entonces, para cada una de las queries puede verificar en las 26 letras si existe alguna en ese rango, haciendo el range query al arreglo correspondiente a la letra que se esta verificando.

Complejidad: $O(26|S| + 26q)$