Olimpiada Informática Española
Soluciones de los problemas

2012

Día 1

Comedor escolar [Ad-hoc]

Problema: Dado el presupuesto de un comedor y el precio de cada uno de los platos, calcular el número de raciones que se pueden servir.

Solución: La solución es la división entera entre el presupuesto y el coste de una ración.

Contar hellos [Implementación]

Problema: Dado un conjunto de palabras, ver cuantas de ellas son "hello".

Solución: Contamos el número de palabras que sean igual a "hello".

Mediana [Sets]

Problema: Dado un conjunto de n palabras distintas, decir en cada momento cual es la mediana de las palabras.

Solución: Una posible implementación sería tener dos sets A, B, de modo que todos los elementos de A sean menores que los de B y, que el primer elemento de B sea la mediana. Cada vez que nos encontramos con una palabra la insertamos en A si es menor que la primera palabra de B, de no ser así, la insertamos en B. En todo momento tendremos que ir moviendo la primera palabra de B a A o la última palabra de A a B, para asegurar que el primer elemento de B sigue siendo la mediana. Este algoritmo tiene un coste de $O(n \log n)$.

Sopa de cartas [Implementación]

Problema: Dada una sopa de letras formada por las cartas $(A, 2, \ldots, 9, J, Q, K)$ marcar en cualquier sentido las secuencias de cartas repetidas o que formen una escalera, hallando además la secuencia más larga.

Solución: Es suficiente con iterar por todas las filas, columnas y diagonales.

Tesoros al noreste [Estructuras de datos]

Problema: Dado un conjunto de n coordenadas (x_i, y_i), en cada una de las cuales hay un tesoro de cierto valor w_i, hallar la máxima suma de valores que podemos obtener, empezando en $(0, 0)$ y moviéndonos solo hacia arriba o hacia la derecha.

Solución: Una posible solución sería realizar un Segment Tree de sumas donde ST[y] es el máximo valor que podemos obtener sin superar la coordenada y. Recorremos los tesoros crecientemente según su coordenada x_i, buscamos cual es el máximo valor que se puede obtener en coordenadas $\leq y_i$ y actualizamos ST[y_i] como ese valor más w_i. Para no tener problemas de memoria es necesario realizar antes una compresión de coordenadas.

El agua de Vallcarca [Binary Search]

Problema: Dada una jarra de capacidad C y temperatura inicial 5º, una secuencia de n instantes en los que beberemos una cierta cantidad de agua, calcular la mínima C de modo que el agua de la jarra esté siempre al menos a 10º, sabiendo que cuando bebemos, rellenamos la jarra con agua de 25º y sabiendo el ritmo de enfriamiento de la nevera así como la temperatura final al mezclar volúmenes de agua a distinta temperatura.

Solución: Una opción es realizar una búsqueda binaria sobre C, comprobando si con esa cantidad se cumplen los requisitos, iterando por las veces que bebemos agua.

Día 2

Camino máximo [Grafos]

Problema: Dado un árbol ponderado, hallar la mayor distancia entre dos nodos (conocida como el diámetro de un grafo).

Solución: Es suficiente hallar el nodo más lejano (v) de uno cualquiera y después hallar el nodo u más lejano a v, la distancia entre estos nodos es el diámetro del árbol. Para hallar el nodo más alejado de un nodo en un árbol, podemos utilizar un DFS.

Carrera de sacos en downtown [Ad-hoc]

Problema: Dados n tramos de carretera, cada uno de cierta longitud l_i y con cierta cantidad de público p_i. Hallar el punto inicial y final de una carrera si queremos que esta dure como mucho L metros y maximice la cantidad de espectadores.

Solución: Para cada tramo t_i encontramos el tramo t_j con $j < i$ de modo que j sea la menor tal que la distancia entre los tramos no supere L. Está claro que si la carrera acabase en el tramo i, la carrera debería empezar en el tramo j. Recorremos pues todos los tramos i, avanzando j cuando la distancia entre los tramos i y j sea mayor a L y actualizando la cantidad de público. La solución será el recorrido que tenga más público.

Mate en uno [Implementación]

Problema: Dado un tablero de ajedrez con la disposición de las piezas y una serie de restricciones, decir si es posible que las piezas blancas hagan mate en la siguiente jugada.

Solución: Simplemente observamos si las piezas blancas pueden amenazar al rey sin que este pueda moverse a una casilla adyacente no amenazada.

Multiset [Estructuras de datos]

Problema: Dado un mulitconjunto vacío implementar las operaciones: 1) inserción, 2) calcular el número de enteros (x) del multiconjunto entre dos números dados ($l \leq x \leq r$) y 3) calcular la suma de los enteros (x) del multiconjunto entre dos números dados ($l \leq x \leq r$).

Solución: La solución esperada sería implementar una estructura de datos que permita realizar esas operaciones en tiempo $O(n \log n)$, como por ejemplo, un Treap o un AVL.

El mejor par [Implementación]

Problema: Dado un conjunto de números, hallar el par de números adyacentes con mayor suma.

Solución: Simplemente vemos cuál de las sumas de números adyacentes es mayor.

Jugando con números [Backtracking]

Problema: Dado un conjunto de n números, hallar los diferentes resultados que se pueden obtener añadiendo sumas o multiplicaciones entre los números y tantos paréntesis como se desee.

Solución: Una posible solución sería hacer un backtracking, implementando una función que dados dos índices (l, r) nos devuelva una lista con todos los números que se pueden obtener realizando operaciones con todos los números entre l y r. Esta función sería recursiva, y para todo m, $l < m < r$ calcularíamos $f(l, m) + f(m, r)$ y $f(l, m) * f(m, r)$.