Tower Of Cubes

Johanna Beltran y Diego Triviño

2012

Índice

1. Introducción

Tower of cubes es un problema de programación dinámina, el cual encontramos en el juez virtual UVA con el número **10051**. Este documento busca mostrar una de las tantas soluciones desde el enfoque matemático, el objetivo es realizar el código con la solución del problema en cualquier lenguaje de programación con la ayuda de este documento.

2. Definición del problema

Este problema consiste en construir una torre con los cubos que estan sujetas a las siguientes restricciones:

1. Nunca se debe colocar un cubo más pesado en uno más ligero.
2. La cara inferior de cada cubo (excepto la parte inferior del cubo, que está tumbado en el suelo) debe tener el mismo color que la cara superior del cubo por debajo de ella.
3. Se debe construir la torre más alta posible.
4. El numero de cubos N debe ser: $1 \leq N \leq 500$.
5. Los datos estrictamente necesarios de cada cubo son los colores de las caras que se ingresan en el siguiente orden: delante, detrás, a la izquierda, a la derecha, arriba y abajo.
6. Los colores C se identifican por números enteros en el rango de: $1 \leq C \leq 100$.
7. La entrada finaliza cuando $N = 0$.
8. Usted puede asumir que los cubos se dan en el orden creciente de sus pesos, es decir, el primer cubo es el más ligero y el cubo N es el más pesado.

2.1. Entrada

La entrada constará de varios casos de entrada. La primera línea de cada caso de prueba contiene un número positivo N que representa el número de cubos a analizar; las siguientes líneas i contienen la descripción de cada cubo es decir los colores de sus respectivos lados e ingresados en el orden ya mencionado anteriormente . El final de la entrada de los datos se indica con el número 0.

EJEMPLO

3

1 2 2 2 1 2

3 3 3 3 3 3

3 2 1 1 1 1

10

1 5 10 3 6 5

2 6 7 3 6 9

5 7 3 2 1 9

1 3 3 5 8 10

6 6 2 2 4 4

1 2 3 4 5 6

10 9 8 7 6 5

6 1 2 3 4 7

1 2 3 3 2 1

3 2 1 1 2 3

0

2.2. Salida

Para cada caso de prueba en la primera entrada debe imprimir el número de caso de prueba en una línea separada, como se muestra en el ejemplo de salida. En la siguiente línea imprimirá el número de cubos en la torre más alta que haya desarrollado. En la línea siguiente se describen los cubos en su torre, de arriba a abajo con una descripción por línea. Cada descripción contiene un número entero (que da el número de serie de este cubo en la entrada) seguido de un único espacio en blanco y luego la cadena de identificación (delante, detrás, izquierda, derecha, arriba o abajo) de la cara superior del cubo en el torre. Tenga en cuenta que puede haber múltiples soluciones y cualquiera de ellos es aceptable.

EJEMPLO ANTERIOR

```
Case  1
2
2 front
3 front

Case 2
8
1 bottom
2 back
3 right
4 left
6 top
8 front
9 front
10 top
```

3. Modelamiento matemático

Para este problema se decidió utilizar 4 matrices:
Cubos: en esta matriz se van a guardar todos los distintos cubos que se ingresan juntos con sus respectivas caras.
TamañoTorres: en esta matriz se van a ir guardando los máximos tamaños que pueden llegar a tener una torre.
MatrizIncidencia: Se encarga de guardar todas las distintas conexiones que puede llegar a tener un cubo con otro, es d
ConexionesCara: Cuando dos cubos se relacionan entre sí, es decir que tienen un color en común, en esta matriz se al

4. Planteamiento de la solución

Una vez se han completado todas las matrices, se busca en $TamañoTorres$ cuál es la más grande, después en la $MatrizIncidencia$ se busca cuáles son los cubos que se encuentran que conforman la torre y para finalizar se revisa en la matriz $ConexionesCara$ cual es color que une dos cubos. Las matrices tienen las siguientes características:

Matriz Cubos: En cada fila de la matriz se almacenan todos los colores un cubo, los cubos se guardan en el mismo orc

1 2 2 2 1 2
3 3 3 3 3 3
3 2 1 1 1 1

La matriz TamanoTorre seria:

$$L = \begin{pmatrix} 1 & 2 & 2 & 2 & 1 & 2 \\ 3 & 3 & 3 & 3 & 3 & 3 \\ 3 & 2 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Matriz TamañoTorres: Esta matriz guarda los distintos tamaños de torres que se pueden generar dependiendo de las distintas combinaciones de cubos. Por ejemplo para el caso anterior su matriz de TamañoTorres sería la siguiente:

$$M = \begin{pmatrix} 2 & 2 & 2 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{pmatrix}$$

MatrizIncidencia: Cada fila de esta matriz representa un cubo y cada uno de sus filas representa el color del cubo, se coloca un 1 para representarlo. Para nuestro ejemplo la $MatrizIncidencia$ es:

$$M = \begin{pmatrix} 2 & 2 & 2 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{pmatrix}$$

ConexiónesCara: Guarda un numero del 1 al 6 representado la cara con la cual se conectan los cubos. Para nuestro ejemplo la matriz conexionCara es:

$$I = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 1 \\ 3 & 2 & 1 \end{pmatrix}$$

5. Conclusiones

1. Para este problema es necesario y conveniente utilizar programación dinámica porque no existe la necesidad de recalcular aquellas soluciones que ya teníamos en algún momento de la ejecución; ahorrando tiempo y codificación en la solución.

2. Para este problema es conveniente utilizar una matriz para guardar los datos que se van encontrando y evitar redundancia.