

MAXIMUM PRODUCT OF TWO ELEMENTS IN AN ARRAY

Nos dan un arreglo de números enteros (num). Se nos pide elegir dos índices (i y j) para regresar el valor máximo de:

$(\text{nums}[i]-1) * (\text{nums}[j]-1)$

*SE USARÁ BUBBLE SORT

| | | | |
|---|---|---|---|
| 3 | 4 | 5 | 2 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 3 | 4 | 2 | 5 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 3 | 2 | 4 | 5 |
|---|---|---|---|

Complejidad:

- $O(n)$: Con cada iteración del algoritmo se hacen comparaciones de valores constantes.

LINK AL VIDEO:

<https://www.loom.com/share/3c2b5aadfb1245af892658b5a98f6a73>

BACKSPACE STRING COMPARE

Nos dan dos string "s" y "t", regresa "true" si ambos tienen la misma "longitud". Al ingresar "#" a cualquiera de los dos strings contará como una especie de eliminación del caracter anterior (como si fuera un pop).

*SE PUEDE RESOLVER CON EL USO DE STACKS

| |
|---|
| C |
| # |
| B |
| A |

Stack S

| |
|---|
| C |
| # |
| D |
| A |

Stack T

Complejidad:

- $O(n+m)$: Depende de la longitud de ambos string,
- Función process: $O(n+m)$ - recorre cada string una sola vez
- operaciones push y pop son $O(1)$
- En el peor caso, no se borra ningún caracter y se guardan todos (sigue siendo $O(n+m)$)

LINK AL VIDEO:

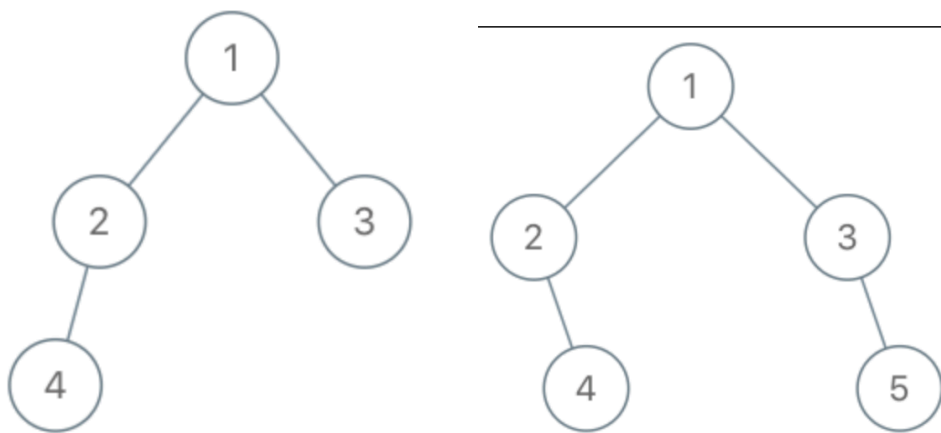
<https://www.loom.com/share/285bc28c2d214182a1510a6898954743>

COUSINS IN BINARY TREE

Nos dan una raíz de un árbol binario, que contiene los nodos "X" y "Y". Se regresa mediante una función booleana si los nodos correspondientes a los valores "X" y "Y" son primos.

*Los árboles binarios son primos si: tienen la misma profundidad con diferentes padres como raíz.

*SE PUEDE RESOLVER CON BFS (Breadth First Search)



Complejidad:

- tiempo $O(n)$: visita cada nodo una sola vez (en el peor de los casos, encuentra X y Y una vez visitados todos los nodos)
- espacio $O(n)$: en el peor de los casos, la cola puede contener todos los nodos de un nivel del árbol, para un árbol balanceado, el nivel más ancho es $n/2$

LINK AL VIDEO:

<https://www.loom.com/share/9d55893259fc476fb0b34beb38a81813>