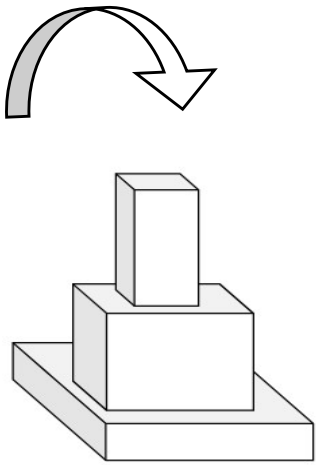
Problema 2)

	Tipo	Descripción
Entrada	Int n Int cajas[0..n)x[0..3): lista de arrays	Cantidad de cajas Lista de arrays de 3 que son las medidas de las cajas
Salida	Int: r	altura máxima de la torre

Tamaño del problema: n

Se usa programación dinámica, generando las tres opciones por cada caja (rotándola) y con estas se recorren los diferentes caminos en orden descendente hasta llegar al final. Al final de esto devuelve el valor más alto.



Complejidad temporal: Primero, el algoritmo crea una lista de cajas en las cuales pondrá cada una de las cajas y sus rotaciones T1=O(n), luego calcula el área de cada caja en la lista que incluye rotaciones T2 = O(n), luego ordena las cajas de mayor a menor T3

= O(nlogn), posteriormente crea una lista de las máximas alturas de cada caja y sus rotaciones T4 = O(n), luego realiza una doble iteración donde compara una caja con las demás para obtener la máxima altura posible, T5 = O(n^2). Para finalizar, busca la mayor altura globalmente T6 = O(n). Como resultado, se tiene T=O(n+n+nlogn+n+n^2+n) = O(n^2)

Complejidad espacial: El algoritmo crea una lista de las cajas y sus rotaciones S1 = O(n), luego crea una lista con la altura de cada una de las cajas de la lista anterior S2 = O(n), crea un par de variables S3 = O(1), entonces S=O(n+n+1)=O(n)