◁ Ejercicio 1 ▷

Se tiene un buque mercante cuya capacidad de carga es de k toneladas y un conjunto de contenedores $c_1, \ldots, c_n$ cuyos pesos respectivos, en toneladas, son $p_1, \ldots, p_n$. Considerando que la capacidad del buque es menor que la suma total de los pesos de los contenedores:

- Diseñe un algoritmo voraz que maximice el número de contenedores cargados.
- Diseñe un algoritmo voraz que intente maximizar el número de toneladas cargadas.

◁ Ejercicio 2 ▷

Un granjero necesita disponer siempre de un determinado fertilizante. La cantidad máxima que puede almacenar la consume en r días, y antes de que eso ocurre necesita acudir a una tienda del pueblo para abastecerse. El problema es que dicha tienda tiene un horario de apertura muy irregular (solo abre determinados días). El granjero conoce los días en que abre la tienda, y desea minimizar el número de desplazamientos al pueblo para abastecerse. Diseñe un algoritmo voraz que determine en qué días debe acudir al pueblo a comprar fertilizante durante un periodo de tiempo determinado (por ejemplo, durante el siguiente mes). Demuestre que el algoritmo encuentra siempre la solución óptima.

◁ Ejercicio 3 ▷

Se deben completar un conjunto de n tareas con plazos límite. Cada una de las tareas consume la misma cantidad de tiempo (una unidad) y, en un instante determinado, podemos realizar únicamente una tarea. La tarea i tiene como plazo límite d_i, y produce un beneficio $g_i > 0$ solo si se realiza en un instante de tiempo $t \leq d_i$. Diseñe un algoritmo voraz que permita seleccionar el conjunto de tareas que asegure el mayor beneficio posible y aplique el algoritmo a la siguiente instancia del problema:

Tarea	i	1	2	3	4
Beneficio	g_i	50	10	15	30
Plazo límite	d_i	2	1	2	1

◁ Ejercicio 4 ▷

Supongamos que disponemos de n ficheros $f_1, f_2, \ldots, f_n$ con tamaños $l_1, l_2, \ldots, l_n$ y un disco SSD de capacidad $d < l_1 + l_2 + \ldots + l_n$.

- Se quiere maximizar el número de ficheros que ha de contener el disco SSD. Para ello, se ordenan los ficheros por orden creciente de su tamaño y se van metiendo ficheros en el disco SSD hasta que no se puedan meter más. Determine si este algoritmo voraz encuentra la solución óptima en todos los casos.
- Se quiere llenar el disco SSD tanto como se pueda. Para ello, se ordenan los ficheros por orden decreciente de su tamaño y se van metiendo los ficheros en el disco SSD hasta que no se puedan meter más. Determine si este algoritmo voraz encuentra la solución óptima en todos los casos.

◁ Ejercicio 5 ▷

El departamento de mercadotecnia de una empresa desea segmentar el conjunto de sus clientes en k grupos para diseñar campañas de mercadotecnia dirigidas a cada grupo por separado. Se supone que se puede caracterizar a cada cliente mediante un vector de características $(c_1, \ldots, c_n)$ y medir la similitud entre dos clientes como la inversa de la distancia entre sus vectores de características. Diseñe un algoritmo voraz que permita agrupar a los clientes de la empresa en k grupos diferentes de tal forma que se maximice la distancia entre clientes de distintos grupos.

◁ Ejercicio 6 ▷

Supongamos que un único camarero debe atender a n clientes de un restaurante y que atender al cliente i-ésimo requiere un tiempo t_i. La satisfacción de un cliente es inversamente proporcional a su tiempo de espera y la propina que recibirá el camarero dependerá de la satisfacción del cliente (si el cliente i espera w_i minutos, la propina que dará al camarero será $p_i = 1/w_i$).

- Diseñe un algoritmo voraz que maximice las ganancias del camarero. Demuestre si el algoritmo diseñado devuelve siempre la solución óptima, o encuentre un contraejemplo que muestre que no lo hace.
- El restaurante toma la decisión de contratar a más camareros para reducir el tiempo de espera de los clientes. Modifique el algoritmo anterior para maximizar las propinas de k camareros.

◁ Ejercicio 7 ▷

Se desea optimizar el uso de las aulas de un centro educativo. Dados un conjunto de aulas y un conjunto de clases con un horario preestablecido (la clase i empieza a la hora s_i y termina a la hora f_i), diseñe un algoritmo voraz que minimice el número de aulas necesario para impartir toda la docencia del centro.

◁ Ejercicio 8 ▷

Suponga que decide montar una empresa de desarrollo de *software*. Para ello, debe adquirir n licencias, cuyo precio actual es de 100 euros cada una. Sin embargo, dispone de un limitado presupuesto que solo le permite adquirir una licencia mensualmente. El problema es que, aunque ahora mismo todas las licencias cuestan lo mismo (100 euros), el precio de la licencia i se incrementa mensualmente un tanto por ciento Δp_i. Esto es, dentro de k meses, la licencia i le costará $100 \cdot (1 + \Delta p_i)^k$ euros. Sabiendo que, al final, necesitará tener todas las licencias en regla, diseñe un algoritmo voraz que minimice el coste de adquisición de las n licencias a partir de sus tasas de incremento de precio Δp_i.

◁ Ejercicio 9 ▷

Se dispone de una memoria caché con capacidad para almacenar k datos. Cuando la caché recibe una solicitud d_i, puede que el dato ya esté en la caché [hit] o que se tenga que buscar en memoria [miss], en cuyo caso se reemplaza alguno de los datos ya almacenados en la caché por el dato que se acaba de obtener. Si se conocen de antemano el conjunto de solicitudes que se deben atender $d_1, \ldots, d_m$, diseñe un algoritmo voraz que minimice el número de fallos de caché.

◁ Ejercicio 10 ▷

Se necesita calcular la matriz producto M de n matrices dadas, $M = M_1 M_2 \ldots M_n$. Por ser asociativa la multiplicación de matrices, existen muchas formas posibles de realizar esa operación, cada una con un coste asociado (en términos del número de multiplicaciones escalares). Si cada M_i es de dimensión $d_{i-1} \times d_i$ $(1 \leq i \leq n)$, multiplicar $M_i M_{i+1}$ requiere $d_{i-1} d_i d_{i+1}$ operaciones. El problema consiste en encontrar el mínimo número de operaciones necesario para calcular el producto M. La siguiente lista presenta cuatro estrategias diferentes:

- Multiplicar primero las matrices $M_i M_{i+1}$ cuya dimensión común d_i sea la menor entre todas, y repetir el proceso.
- Multiplicar primero las matrices $M_i M_{i+1}$ cuya dimensión común d_i sea la mayor entre todas, y repetir el proceso.
- Realizar primero la multiplicación de las matrices $M_i M_{i+1}$ que requiere menor número de operaciones $(d_{i-1} d_i d_{i+1})$, y repetir el proceso.
- Realizar primero la multiplicación de las matrices $M_i M_{i+1}$ que requiere mayor número de operaciones $(d_{i-1} d_i d_{i+1})$, y repetir el proceso.

Determine si alguna de las estrategias propuestas encuentra siempre la solución óptima.