# Práctica Número 4
# Programación Dinámica

## Objetivo

Se trata de implementar un algoritmo que permite calcular el número óptimo de multiplicaciones y la parentización adecuada para multiplicar n matrices suponiendo que se conoce el vector de dimensiones de las matrices a multiplicar.

## 1. Introducción

Existe un algoritmo de programación dinámica expuesto en el material de teoría de la asignatura que permite resolver el problema mencionado. Si el número de matrices a multiplicar es $n$ se puede calcular con coste espacial $O(n^2)$ y coste temporal $O(n^3)$ tanto el número óptimo de multiplicaciones como la parentización adecuada.

## 2. Descripción del Algoritmo

Se consulta en el material de teoría de asignatura, en el tema dedicado a programación dinámica.

## 3. Actividad a realizar

Debe codificarse en Java el algoritmo presentado en el material de teoría. Deberá suministrarse una tabla de experimentos para cada uno de los distintos valores de $n > 2$ (el número de matrices a multiplicar). Se hará 1 tabla, con columnas $n \in \{3, 4, 5, 6, 7, 8\}$. Cada fila contendrá vectores aleatorios de tamaño $n + 1$, donde las componentes son números naturales generados en un rango $[2....M]$. La componente $T[d_0, d_1, ...., d_n][i]$ indica el número óptimo de multiplicaciones para multiplicar $i$ matrices de las dimensiones indicadas por el subvector $[d_0, d_1, ..., d_i]$. En la experimentación pueden considerarse tantas filas como columnas.

En esta parte se entrega el código y la tabla de experimentos.

## 4. Actividad complementaria

Se trata de proponer una estrategia voraz para el problema considerado. Las estrategias voraces, en general, no son óptimas. En el caso de la que se proponga se valorará una demostración de su optimalidad o, en caso contrario, mostrar ejemplos de que se ponga de manifiesto que es subóptima.

La solución propuesta contará de los siguientes elementos:

- Una especificación del problema (basta hacerlo para el cálculo del número de multiplicaciones).
- Una resolución completa en pseudocódigo trasladable a lenguaje de programación indicando claramente la selección voraz.
- Una demostración de optimalidad en el caso de que se encuentre.

- Una implementación en Java.
- Una tabla de experimentos comparando los resultados con el valor óptimo (obtenido mediante el programa de la solución anterior). Usar la misma tabla de la parte anterior.

Se entrega informe con cada uno de los items anteriores.