

# Lab03: Caminos

## 1. Camino entre dos nodos de un grafo.

El propósito de este ejercicio es implementar una función **recursiva** que encuentra un camino entre dos nodos de un grafo que no pase dos veces por un mismo nodo. La estrategia para realizar dicha función se conoce como DFS (*Depth-First Search*).

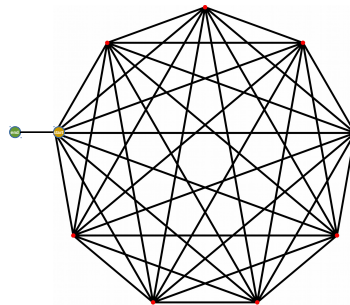
En este ejercicio dispones del fichero (`lab03_path_finder.py`), donde tienes que terminar de implementar la función `find_path`, que dado

- un grafo de entrada;
- un nodo inicial;
- un nodo final;
- y el camino `path` recorrido hasta el momento;

si existe un camino entre el nodo inicial y el final pasando por un nodo  $c$ , debe devolver `path` aumentado con  $c$ . En otro caso debe devolver la lista vacía.

## 2. Coste de tu algoritmo.

Piensa cómo se comporta tu solución al problema anterior cuando se recibe como entrada un grafo que consta de  $n$  nodos rojos, un nodo inicial marrón y un nodo final verde tal y como aparece en la siguiente figura:



¿Cuál es el coste de tu algoritmo en el caso peor? ¿Es exponencial o es polinómico?

## 3. Solución para la versión decisional.

Ahora debes implementar una función más sencilla que sólo decide si existe un camino entre dos nodos de un grafo, sin devolver el camino. El coste de esta función deberá ser polinómico.

En el fichero `lab03_path.py` debes programar la función `graph_has_a_path` que, dado un grafo y los nodos inicial y final, devuelve `True` (cierto) si en dicho grafo hay un camino desde el nodo inicial al final y `False` (falso) si no lo hay. Utiliza la estrategia que se conoce como BFS (*Breadth-First Search*).