Paradigmas de Programación
Práctica 10

Dado un grafo dirigido con ***n*** nodos (numerados de ***0*** a ***n-1***) con pesos en las aristas, se puede usar una matriz ***n x n*** para representar las aristas y sus pesos. Supongamos que los pesos de las aristas están representados por valores enteros (no negativos). Esta matriz podría representarse en *OCaml* con un valor ***w : int option array array***. De este modo si ***w.(i).(j)*** es ***None*** eso significaría que no existe arista del nodo ***i*** al ***j,*** mientras que si es ***Some n***, existiría tal arista y su peso sería ***n***.

Se trata de implementar una función
dijkstra : int option array array -> int option array array, de modo que ***dijkstra w*** sea un valor de tipo ***int option array array*** que represente un grafo que indique si existe o no camino de un vértice a otro del grafo ***w*** y, si es el caso, cuál es el peso mínimo de ese camino.

La función ***dijkstra*** no debe modificar en modo alguno el vector que recibe como argumento, y su comportamiento no debe depender de ningún valor externo a la función (aparte de su propio argumento). Debe comprobarse que la matriz sea cuadrada y que no contiene valores negativos (para que pueda aplicarse el *algoritmo de Dijkstra*), en caso contrario debe activarse la excepción ***Invalid_argument "dijkstra"***.

Para implementar este algoritmo se necesita tener alguna manera de representar una cola de prioridades mínimas. Aunque una simple lista de pares *(prioridad, valor)* podría servir para este fin, resultaría más conveniente disponer de una estructura más eficiente. El módulo ***MinPriQueue***, cuyo código fuente se adjunta a este enunciado, proporciona una implementación funcional de estas colas basada en montículos binarios, que resulta bastante eficiente. Puede compilar el código fuente de este módulo (`ocamlc -c minPrioQueue.mli minPrioQueue.ml`) y cargarlo en el compilador interactivo *ocaml* con el comando `#load "minPrioQueue.cmo"`. En el archivo *minPrioQueue.mli* puede verse la interfaz de este módulo que contiene sólo un tipo de dato abstracto para representar las colas y tres valores de significado bastante obvio. Naturalmente, estos valores pueden usarse, si se desea, en la definición de la función ***dijkstra***.

Escriba la definición de la función ***dijkstra*** en un archivo ***dijkstra.ml*** que debe compilar sin errores con la orden

```
ocamlc -c dijkstra.mli dijkstra.ml
```