Partie IV : jeux à un joueur

On appelle jeu à un joueur un triplet (E, i, F, s) où E est un ensemble d'éléments appelés états, i un élément de E appelé état initial, F un ensemble d'éléments de E appelés états finals et $s : E \to \mathcal{P}(E)$ une application. Une solution de ce jeu est une suite finie $(s_0, \cdots, s_n)$ telle que $s_0 = i$, pour tout $1 \leq i \leq n$, $s_i \in s(s_{i-1})$ et $s_n \in F$.

A un tel jeu est associé le graphe de sommets E et dont les arcs sont les (s_i, s_j) tels que $s_j \in s(s_i)$.

Une façon de trouver une solution est de parcourir le graphe jusqu'à trouver une solution.

On suppose donné un type `état`, une fonction `s : etat -> etat list` et une fonction `final : etat -> bool` qui permet de savoir si un état est final.

Question 1
Ecrire une fonction `bfs1 : int -> bool` qui effectue un parcours en largeur jusqu'à la profondeur m au plus. Elle renverra un booléen indiquant s'il existe une solution à profondeur m.

Question 2
Ecrire une fonction `bfs : unit -> int` qui renvoie la profondeur minimale à laquelle il existe une solution.

Question 3
Ecrire une fonction `dfs : unit -> int` qui renvoie la profondeur minimale à laquelle il existe une solution en effectuant un parcours en profondeur.

On va appliquer ce qui précède au jeu du taquin. Ce jeu s'appuie sur une grille 4×4 contenant une case vide et des cases numérotées de 1 à 15. On peut avancer toutes les cases adjacentes à la place de la case vide. Un état du jeu du taquin sera donc une grille 4×4. La position d'indice (i, j) contiendra le numéro situé sur cette case. Le numéro de la case vide pourra être un numéro arbitraire.

Question 4
Définir un type `etat`, des fonctions `final` et `s` correspondant au jeu du taquin. En déduit, à l'aide de la question 2 ou 3, une fonction résolvant le jeu du taquin.