Université de Bordeaux
Année 2024-2025

Licence Informatique, 3e année

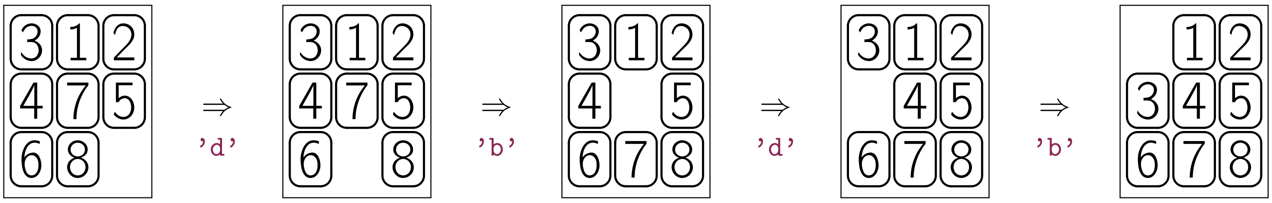
Techniques Algorithmiques et Programmation

TP noté – 2h10

Taquin Game (Sliding Puzzle)

Consider the Taquin Game (Sliding Puzzle) of order n, a single-player game composed of $n^2 - 1$ numbered tiles from 1 to $n^2 - 1$ that slide in an $n \times n$ frame, with one empty space. The goal is to arrange all the tiles in order from any initial configuration. A tile can only move if it is adjacent to the empty space (4-neighborhood). Once moved, the new configuration has a different empty space.

Below is an example of a Taquin Game configuration of order $n = 3$ (left figure), where four moves allow winning, i.e., reaching the winning configuration (right figure).



A configuration is represented by a 2D array where `G[i][j]` is a unique integer in $[0, n^2 - 1]$, with the empty space coded as 0. The cell `G[i][j]` corresponds to the tile located in the i-th row and j-th column, with `G[0][0]` being the top-left cell. It is also the empty space in the winning configuration.

Depending on the initial configuration, it is not always possible to reach the winning configuration. For example, if the initial configuration is the winning configuration with tiles numbered 1 and 2 swapped, it is impossible to win.

Each move is determined by its direction towards the empty space. Specifically :

`'d'` : move a tile to the right (→ : increment j),
`'g'` : move a tile to the left (← : decrement j),
`'b'` : move a tile downwards (↓ : increment i),
`'h'` : move a tile upwards (↑ : decrement i).

Note that a move changes two values in the array `G` : the value of the moved cell and the value of the cell containing 0, which represents the empty space.

A sequence of moves is encoded by a string, such as `"dbdb"` corresponding to the four moves in the example. Some moves are invalid, such as `'d'` or `'b'` for the winning configuration, as its empty space has no adjacent tile that can move to the right or down.

The goal of the assignment is to program the functions used to find the shortest sequence of moves to win, when possible. This search will be done using the A* algorithm seen in class and tutorials. For this, carefully read the instructions in the comments in the file `taquin.c`, edit this file, test your functions, and then submit only this completed file before the end of the session on the course Moodle page.

FIN.