Formulation of the Temple Trap Puzzle as a Search Problem

Scope

This document formally **formulates the Temple Trap puzzle as a search problem**. It specifies the world structure, the tile set, the state representation, the Pawn's move, the lock rule, and the goal test. Finally, it presents an assignment to be solved using standard search algorithms.

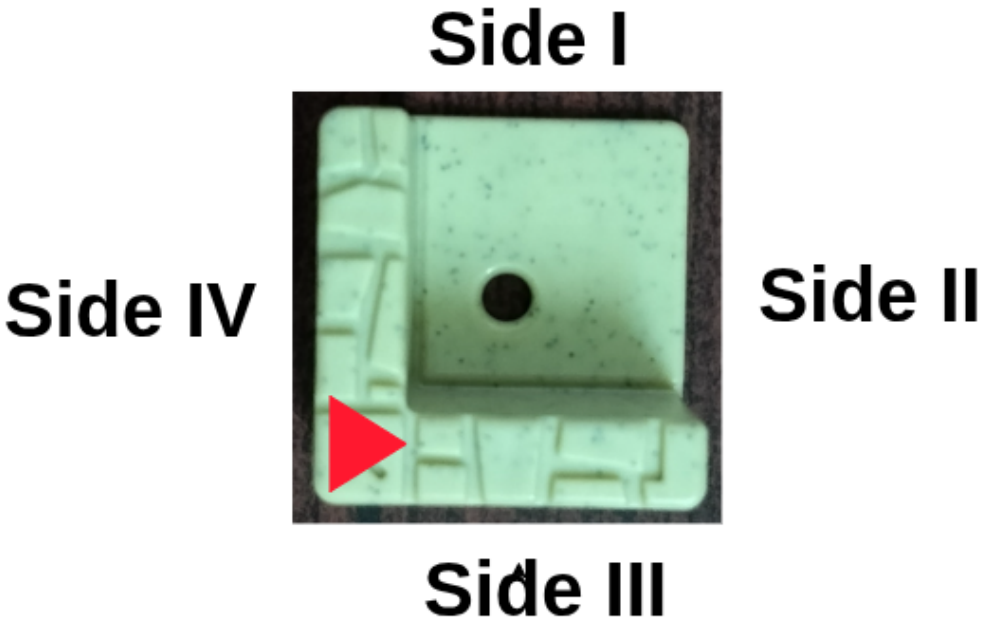

Figure 1: Sides of a Tile Example.

Side legend (unique labels)

To avoid directional ambiguity, each tile is always oriented such that its unique identifying symbol lies in the **bottom-left corner**. With this fixed orientation, we label the four sides of a tile in a clockwise manner as:

Side I, **Side II**, **Side III**, **Side IV**,

where Side I is the edge at the top, Side II is the edge on the right, Side III is the edge at the bottom, and Side IV is the edge on the left.

Two adjacent tiles are considered connected on a layer if and only if:

- **Horizontal connectivity:** the left tile opens Side II and the right tile opens Side IV.
- **Vertical connectivity:** the upper tile opens Side III and the lower tile opens Side I.

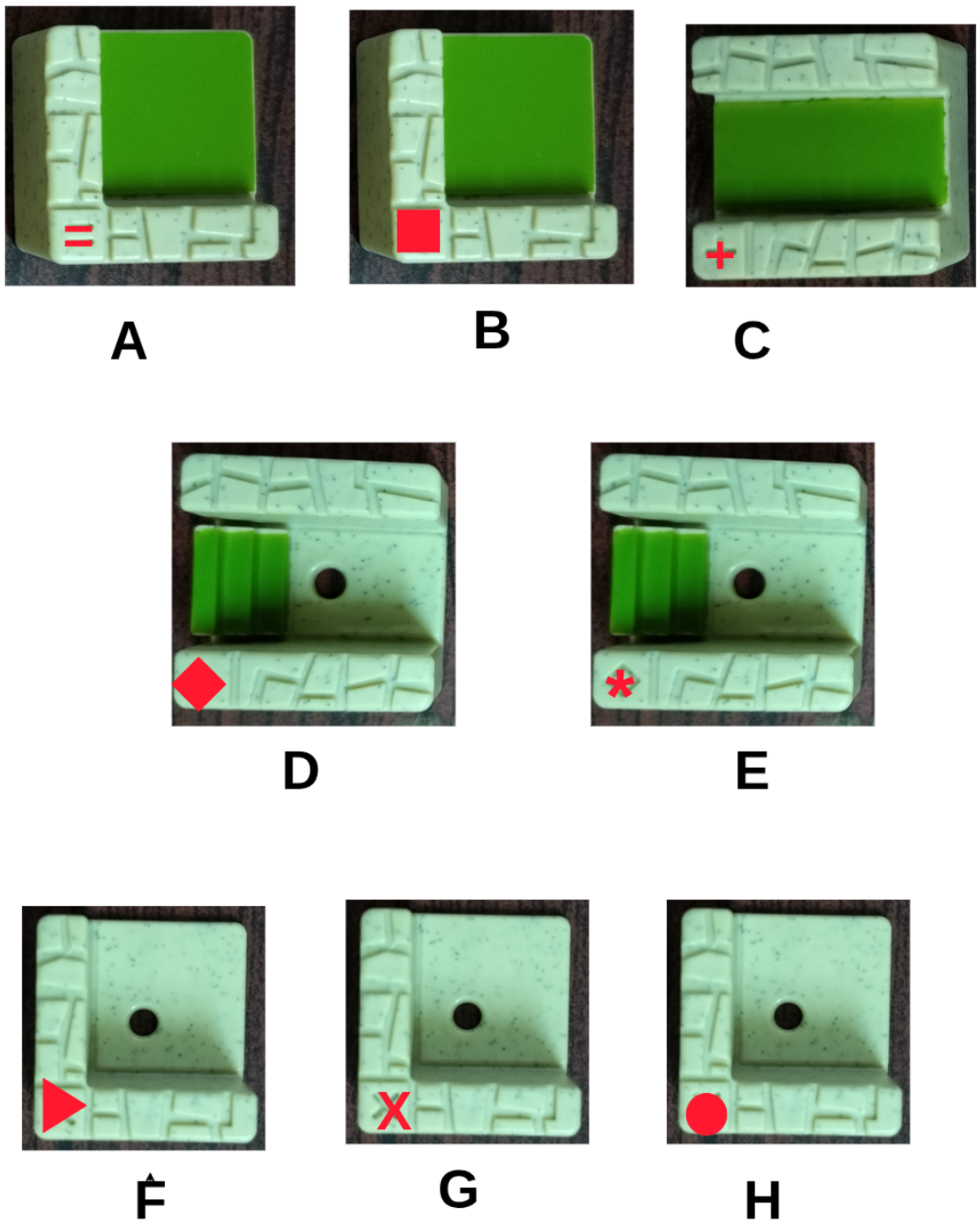

Figure 2: Temple Trap tiles (A–H) with symbols in the bottom-left corner.

Tiles

Exactly these eight distinct tiles and orientations:

Top-only corridors

- **A:** Top opens = {Side I, Side II}; Ground opens = none; Hole: No; Stairs: No
- **B:** Top opens = {Side I, Side II}; Ground opens = none; Hole: No; Stairs: No
- **C:** Top opens = {Side I, Side III}; Ground opens = none; Hole: No; Stairs: No

Stairs with holes

- **D:** Top opens = {Side IV}; Ground opens = {Side II}; Hole: Yes; Stairs: Yes (stair on **Side IV**)
- **E:** Top opens = {Side IV}; Ground opens = {Side II}; Hole: Yes; Stairs: Yes (stair on **Side IV**)

Bottom-only hole corners

- **F:** Top opens = none; Ground opens = {Side I, Side II}; Hole: Yes; Stairs: No
- **G:** Top opens = none; Ground opens = {Side II, Side III}; Hole: Yes; Stairs: No
- **H:** Top opens = none; Ground opens = {Side III, Side IV}; Hole: Yes; Stairs: No

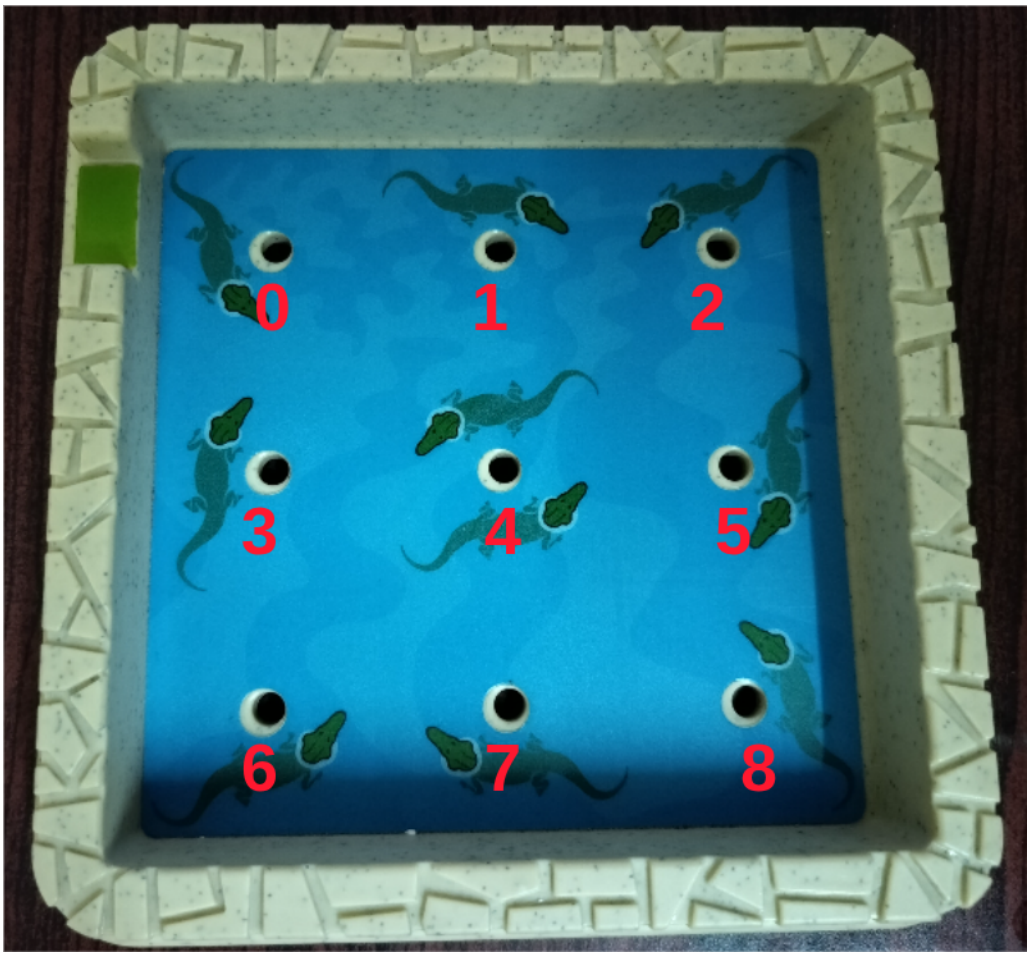

Figure 3: The Cells in the board.

Grid and exit

Grid: 3×3 cells indexed row-major:

$$\begin{array}{ccc} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{array}$$ (exactly one cell is the blank "␣").

Floors: The board consists of two layers: GROUND and TOP. A change of floor occurs only when a Pawn moves onto a tile containing stairs.
Exit doorway (fixed): The exit is located at cell **0**, on its **left boundary**. Hence, the tile placed in cell 0 must be oriented such that its edge coinciding with the left side of cell 0 is open, thereby allowing passage out of the board.

State Space

State B: The puzzle is represented as a 3×3 board layout, consisting of the eight tiles (A–H) plus one empty position "_".

The solver is provided with:

- The placement of each tile (A–H) given as its current cell index (from 0 to 8).
- The position of the Pawn, specified by its current cell index (from 0 to 8).

Walk

Given a board B, the Pawn can move if:

- **Ground:** Two side–by–side cells connect if both touching sides are open on GROUND.
- **Top:** Same rule, but for openings on TOP.
- **Stairs:** In a cell with tile D or E (e.g., tiles with stairs), the Pawn may switch between GROUND and TOP.
- **Blank:** The Pawn can never enter the blank "_" cell.

Pawn's Move, Lock Rule, Cost

Pawn's Move: Before any slide, the Pawn may freely walk in the region reachable from its current cell. This includes movement on the GROUND layer, climbing stairs, and traversing connected TOP tiles. The Pawn may only rest on tiles that contain a hole.
Lock rule: A tile with a hole can slide into the blank only if *the Pawn is not currently located on that tile.*
Cost: Sliding one tiles has cost 1 and Pawn movement has also a cost of 1.

Goal State

The goal is reached if there exists a valid walk from the Pawn's current position to cell 0, *and* the edge of the tile placed in cell 0 that coincides with the left boundary of cell 0 is open, allowing the Pawn to exit the board.

An Example from Initial to Goal State

An example solution can be illustrated by starting from an initial configuration of the 3×3 board (with tiles A–H and one blank), moving the Pawn by valid walks, and performing legal slides.
Through a sequence of such moves, the Pawn eventually reaches cell 0, where the tile at cell 0 opens
to the left boundary, enabling the Pawn to exit the temple.

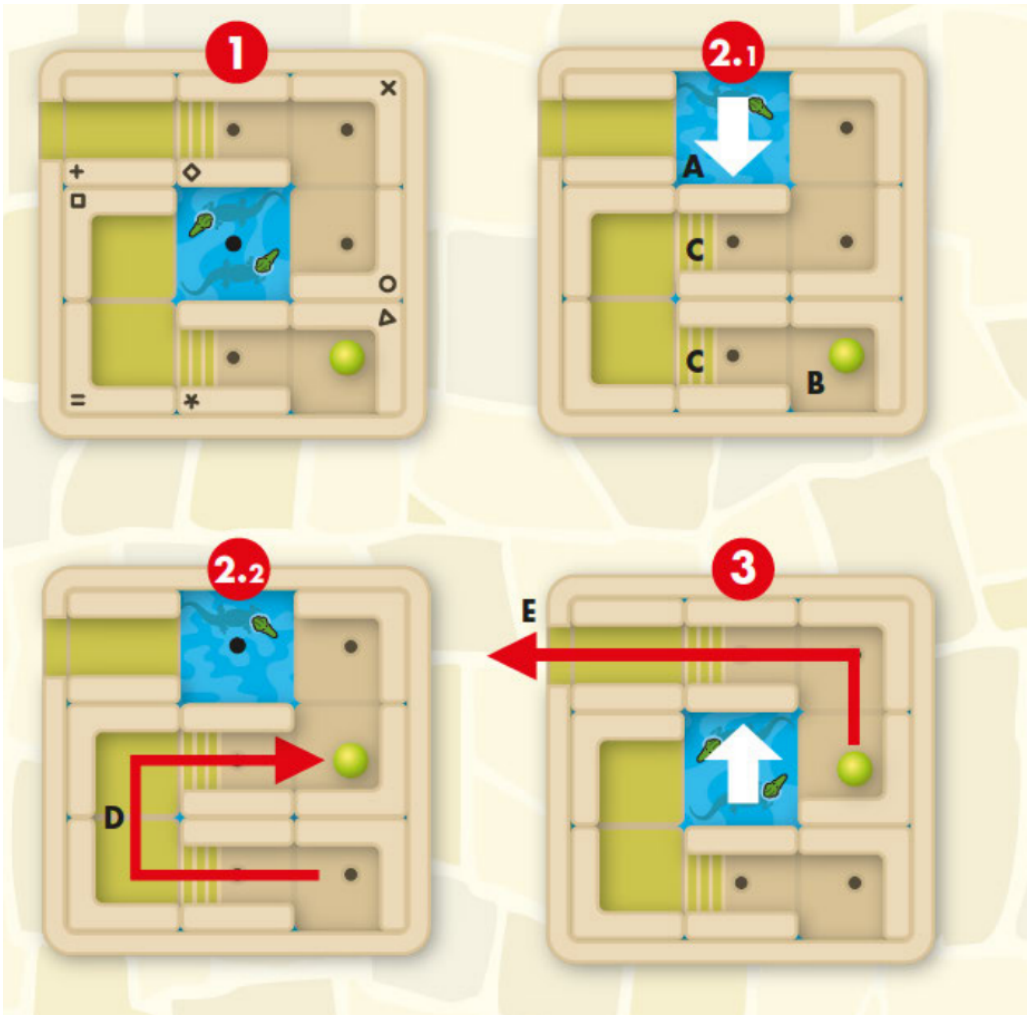

Figure 4: An Example

Assignment

Solve the initial boards shown in Figures 5, 7, and 9. You are required to use a search algorithm of your choice and minimise the total cost. The optimal solutions are provided in Figures 6, 8, and 10 for validation of your results.

Requirements:

- Define the State Space.
- Define the Action Space.
- Specify the heuristic function(s), if any.
- Submit runnable code. Test cases will be provided, and the algorithm must generate the correct solution.

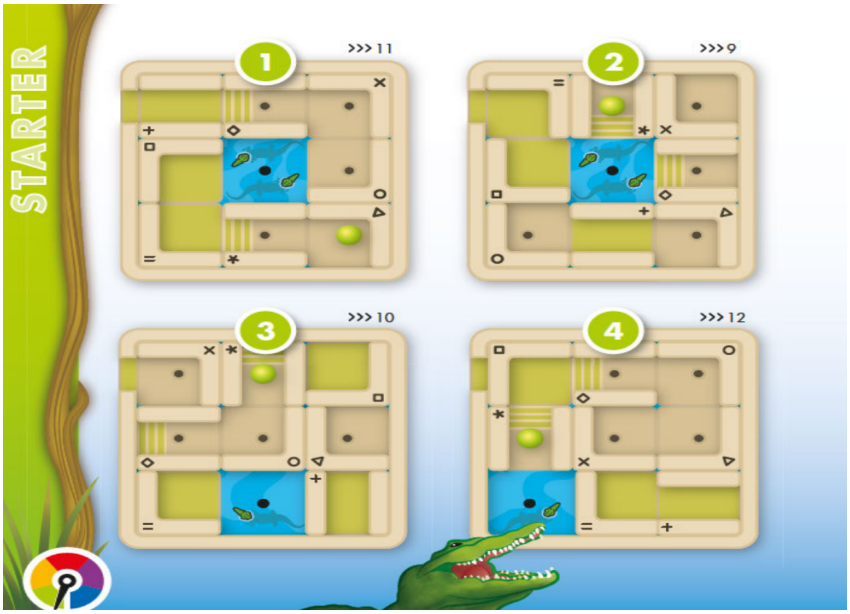

Figure 5: Level 1

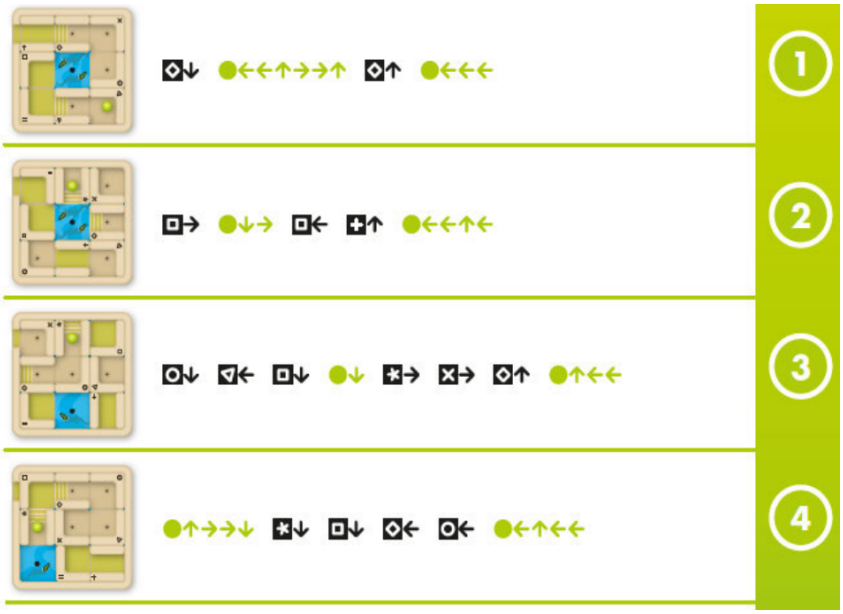

Figure 6: Solution to Level 1

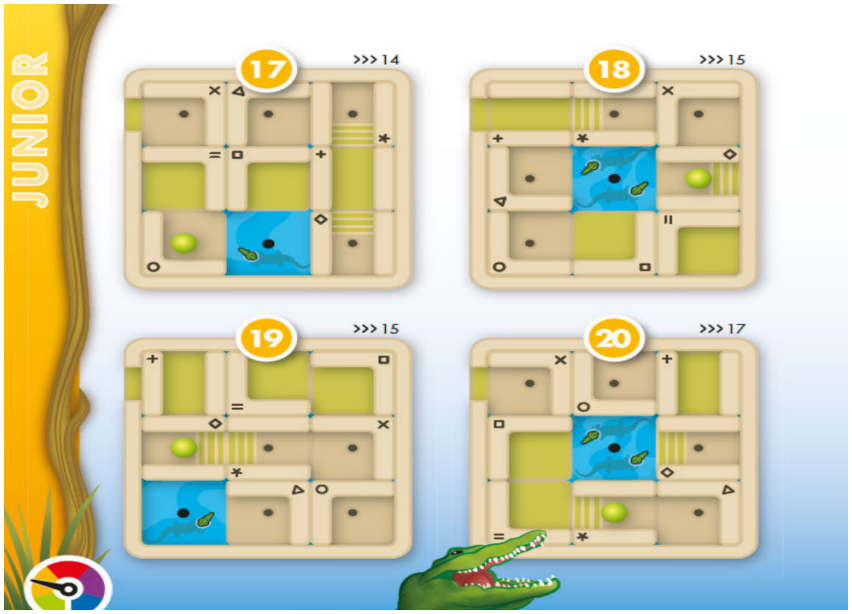

Figure 7: Level 2

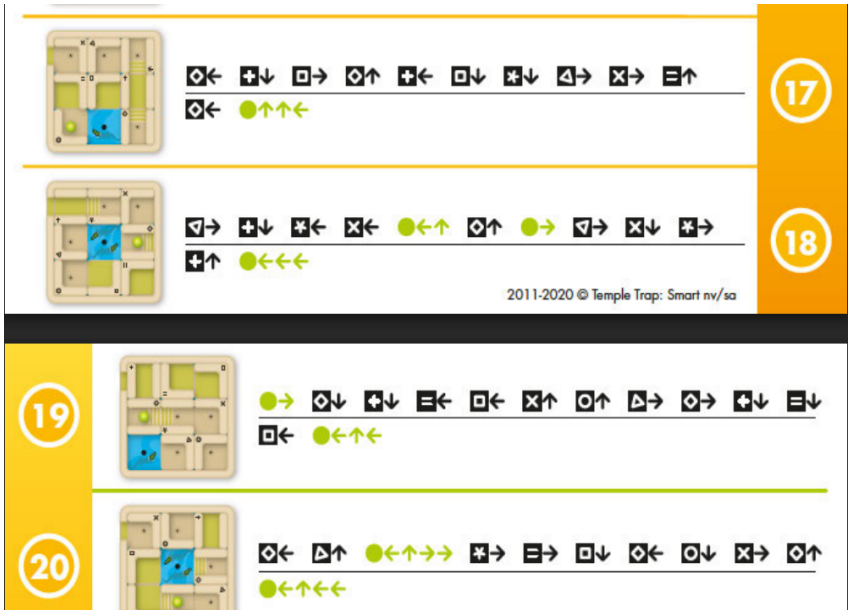

Figure 8: Solution to Level 2

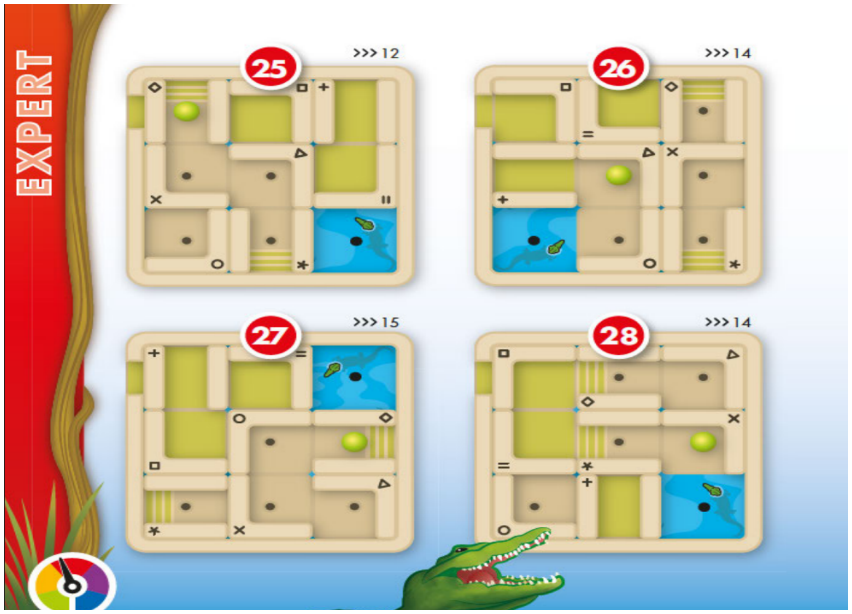

Figure 9: Level 3

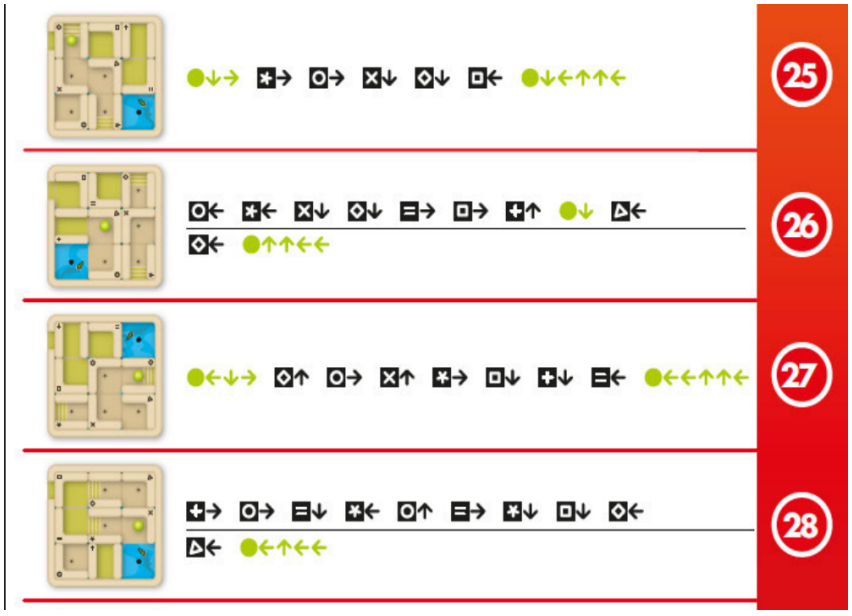

Figure 10: Solution to Level 3