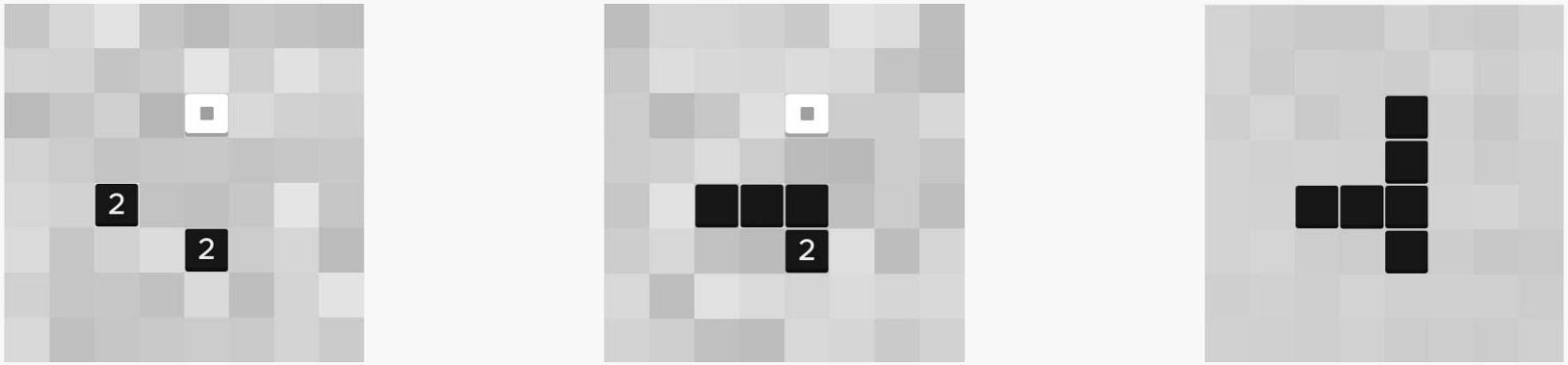
IART – CHECKPOINT 1

Daniel Brandão – 201705812
Pedro Moás – 201705208
Gaspar Pinheiro - 201704700

ZHED - Single Player Game



- ZHED is a grid based puzzle game where in order to complete each level a numbered cell must be expanded to reach the goal cell.
- Each numbered cell can be expanded in one of four directions and overlapped. Each cell expands *n* cells in the direction chosen, decreasing by one for each empty cell. When a expanding cell overlaps an already filled cell, the number of cells to be filled in the direction of the expansion is not decreased.

Related Work

- https://www.wilgysef.com/articles/zhed-solver/
- https://github.com/WiLGYSeF/zhed-solver
- https://www.cin.ufpe.br/~if684/EC/aulas-IASimbolica/korf96-search.pdf
- http://archive.oreilly.com/oreillyschool/courses/data-structures-algorithms/singlePlayer.html
- http://www.pvv.ntnu.no/~spaans/spec-cs.pdf

Problem Formulation

For each level, the puzzle size and the numbered cells' positions are diferent. Therefore we are using as an example for the formulation a puzzle size of 4.

- State Representation: NxN Matrix (List of Lists of integers, N = Puzzle Size), where each cell can have a value, *Val*, of:
  - A positive number, representing the expandable length of the cell;
  - 0, representing an empty cell;
  - -1, representing an expanded cell;
  - -2, representing the goal cell.
  - -3, representing the reached goal cell.
- Initial State: Matrix with at least one Val = -2 , at least one Val > 0, and no Val = –1 or Val = –3 cells
- Final State: Matrix with one Val = -3

1	0	-2	0
0	0	0	0
0	0	0	3
0	2	0	0

Initial State example

-1	-1	-3	0
0	-1	0	0
-1	-1	-1	-1
0	-1	0	0

Final State example

Operators

Operator	Pre-conditions	Effects	Cost

Move [X,Y] Up	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X, Y – i) != 0) continue;
		Board(X, Y – i) = -1; i++;
			1

Move [X,Y] Down	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X, Y + i) != 0) continue;
		Board(X, Y + i) = -1; i++;
			1

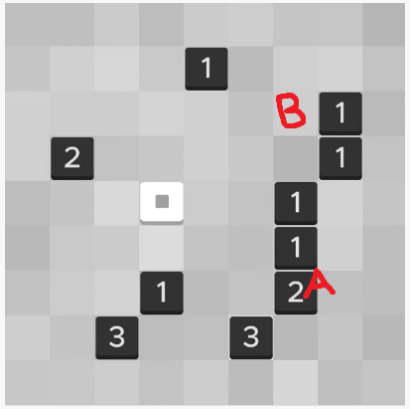
Move [X,Y] Left	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X - i, Y) != 0) continue;
		Board(X - i, Y) = -1; i++;
			1

Move [X,Y] Right	Board(X,Y) > 0	i = 1;
		While (i <= Board(X, Y))
		If (Board(X + i, Y) != 0) continue;
		Board(X + i, Y) = -1; i++;
			1

Heuristics

- We plan on implementing several search algorithms, such as breadth-first, depth-first, greedy, A*, and comparing the results we achieve.
- For the heuristic methods (greedy, A*), we will try different heuristics, such as:
  - H1 = Minimum *Zhed Distance* between a Value Cell and a Finish Tile.
  - *H2 = (1 - Number of Reached Finish Tiles) / (Number of tiles aligned with a Finish Tile)*
  - *H3 = (1 - Number of Reached Finish Tiles) / Sum(Maximum Tile Reach)*
  - *Hx = A combination of previous heuristics*

The Zhed Distance between a Value Cell and a Tile, **only applicable when they are in the same row or column**, consists of:
(The actual distance between them) -
(The number of used tiles between the) -
(The Cell's Value)



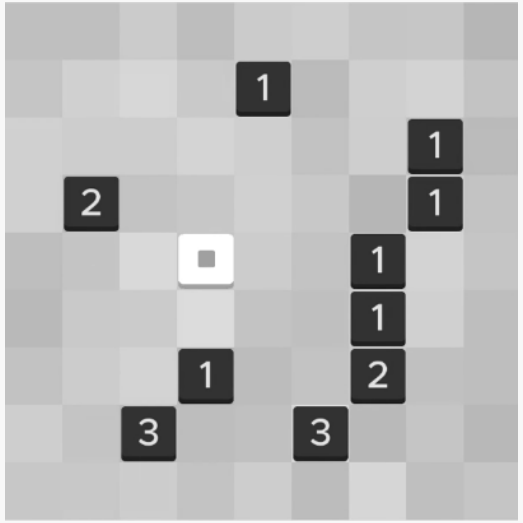
For instance, in this case, the *Zhed Distance* between A and B is 0, as it can reach the destination with no extra blocks between the path (4 – 2 – 2 = 0)

Development

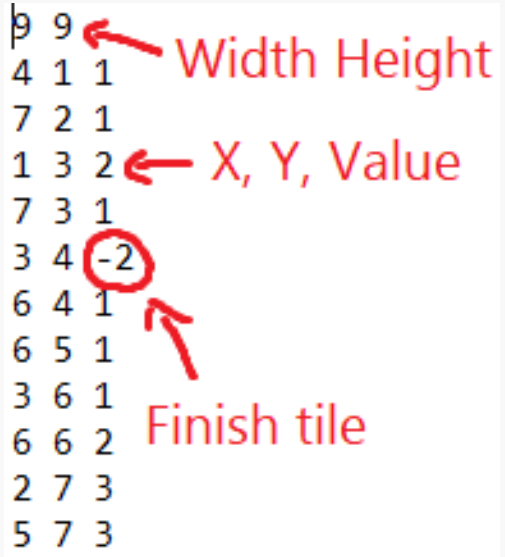
- We are using C# as a programming language
- We have already developed the game's logic and got a working console version, where the user can play a predefined level, or one imported from a file,

For Zhed Level 41, its file structure and console representation are shown below:

Zhed App



File representation



Console representation: