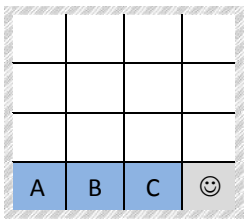
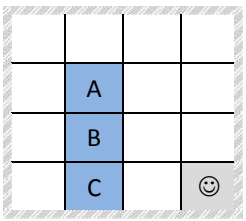
“Blocksworld tile puzzle”

AI Coursework assignment

An ‘agent’ moves in a simulated NxN grid world with the goal of building towers of blocks. Each grid space contains either a ‘tile’ or the agent. Some tiles have letters on them – these are the ‘blocks’. All the other tiles are white. The agent moves up/down/left/right (except where borders prevent it). As the agent moves, the tile that they move onto slides under them into the position that they just came from (see 8-puzzle)[1]. Some examples of moves are given overleaf. The exact start state and goal state for the assignment is shown below. The goal is to build a tower, with these exact blocks in these exact positions as shown. The position of the agent at the end doesn’t matter.



Start state.



Goal state.

Implement the following types of search to (try to) solve this problem: depth first, breadth first, iterative deepening, A* heuristic search. You might consider randomising the ordering of node expansion in depth first. Provide evidence of these methods running/the solutions (action sequences) that they find - provide the shortest action sequence for each method (assuming it completes), from the start state to the goal. Not all methods will necessarily be able to solve this problem – you might experiment with making it easier – e.g. by making the start state closer to the goal state. (If you alter the start state, say what it is when you provide your solutions).

Then examine how the computational time (number of nodes expanded) to reach a solution increases/scales-up with problem size/difficulty. You might control problem difficulty by controlling the depth of the solution (i.e. how far the start state is from the goal state - recommended), or by altering the number of blocks in the problem (i.e. number of non-white tiles), or the size of the grid world, for example.

Produce a figure with problem difficulty on the x-axis and number of nodes expanded to find a solution on the y-axis, and plot results for all search methods (up to the point when they fail). In some cases it is necessary to run the method many times and take an average.

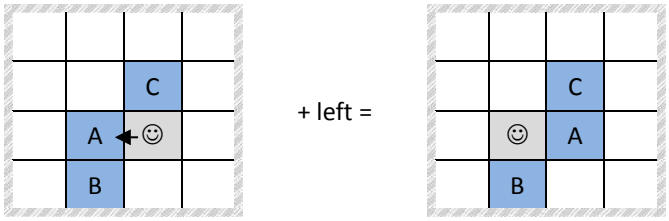
If you find that too easy, get creative. For example, can you still solve it if the world has obstacles (i.e., tiles that cant be moved).

See assignments page for what to hand in/format/headings.

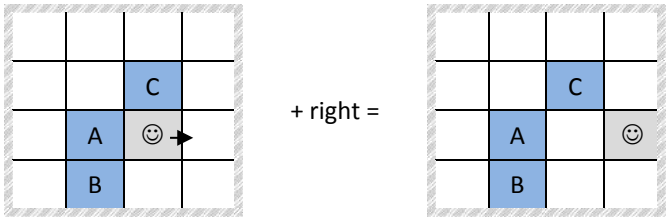
[1] This means that ‘blocks’ can hang in the air – there’s no ‘gravity’ in this world.

Some examples of moves –

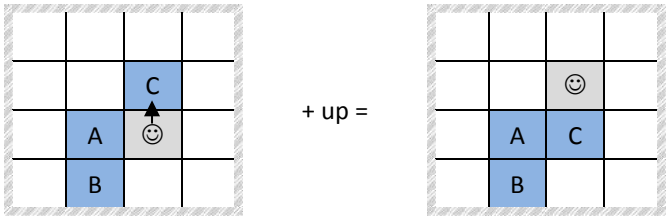
i)



ii)



iii)



iv)