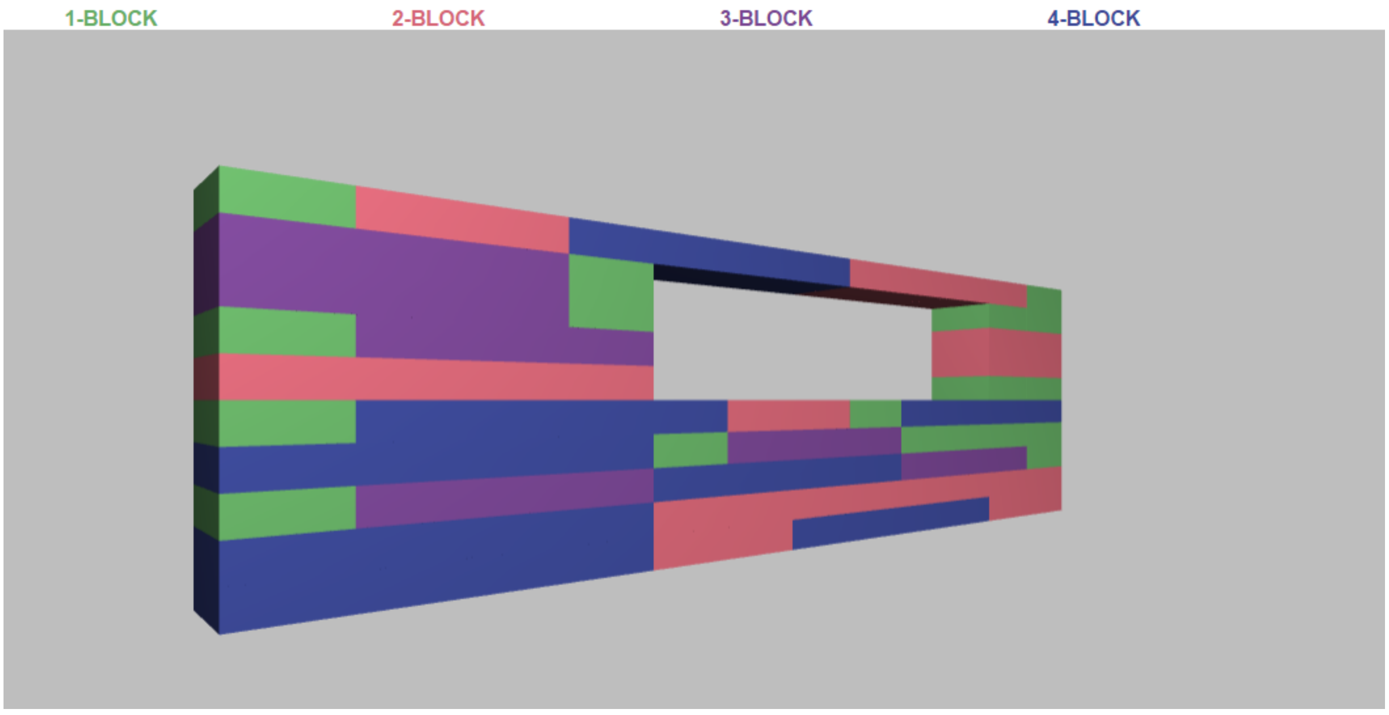
rapidBiz Apps

Assignment Documentation



R D Teja
(AE12B025)

Minimum Challenge: Finding number of ways of ways to arrange without a window and Lego colors

Searches:

I have seen a similar construct in one of my previous placement tests and I knew this can be solved using dynamic programming problem and the algorithm is available on stackoverflow.com. Query “lego blocks dynamic programming” straight away took me to “http://stackoverflow.com/questions/15424945/lego-blocks-dynamic-programming” where the algorithm is clearly discussed.

Algorithm:

First find total possible arrangements of wall (that might violate vertical cut property) and subtract the ones that violate the given properties.

Finding total possibilities:

First find the ways for single row R[1,W], later it can extend it to multiple rows. Row with zero (1 way of choosing nothing) and one column has one possibility i.e. R[1,0] = R[1,1] =1. There are four ways we can construct R[1,i]:

1. Add 1-Block to R[1,i-1]
2. Add 2-Block to R[1,i-2]
3. Add 3-Block to R[1,i-3]
4. Add 4-Block to R[1,i-4]

So, if we know R[1,i-1]...R[1,i-4], R[1,i] would be

$$R[1,i] = R[1,i-1] + R[1,i-2] + R[1,i-3] + R[1,i-4]$$

Since every row is independent of each other, number of ways to construct wall of H x W is

$$A[H,W] = R[1,W]^H$$

Possibilities that violate the vertical cut property:

To enumerate these possibilities, count the possibilities in which the property is first violated after 'k' columns where k can go from 1 to W -1, let it be B[H,k]. Summing up B from k = 1 to W-1 would give the required possibilities. Let S[H,W] holds the number of ways for the structurally connected wall. B[H,k] is connected till k and violates after k, which would give

$$B[H,k] = S[H,k]*A[H,W-k]$$

Net Possibilities:

$$\text{So, } S[H,W] = A[H,W] - \text{sum of } (B[H,k]; \text{ where } k = 1 \text{ to } W-1)$$

Programming:

Programming for computing ways is done in C++ (source: solver.cpp, binary: binary.exe). Input and output is STDIO and input format is as given in the problem statement except that the window height X, Y is 0.

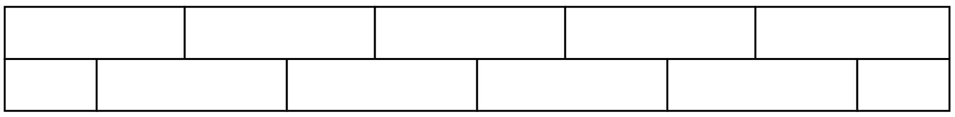
Medium Challenge: Build wall with Lego colors and without windows

Since each size block has same color, adding colors will not change the number of possibilities. For visualizing the wall with colors, we need at least one Lego arrangement which is structurally connected. Above algorithm just counts the number of ways and fails to give anything about the arrangement of blocks.

Algorithm to construct a possible solution:

The algorithm I have come up with is very straightforward, approach is to construct just two rows that are structurally connected (considering number of rows is more than 2) and add the remaining rows with any arrangement (same number of columns) to it and still the wall will be structurally connected.

Constructing two rows with structural connectedness is simple. For first row, starting with 2-Block fill the row with 2-Blocks only (if columns are odd add 1-Block at the end). Whereas for the second row, start with 1-Block and fill the remaining with 2-Blocks (if necessary add 1-Block at the end). It is simple to see that these two rows are connected (see fig. 1) irrespective of number of columns.

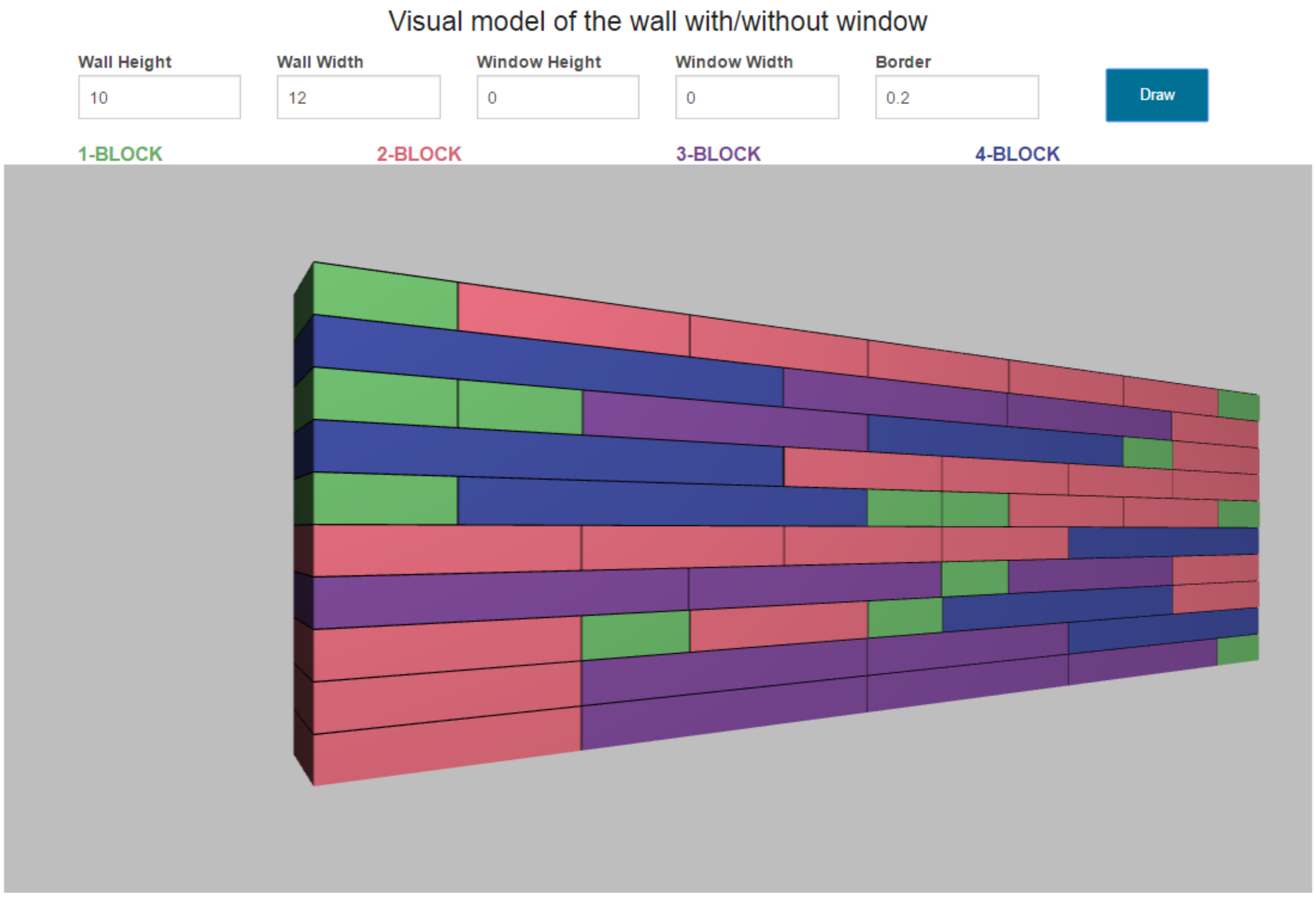


To make it more random, we can replace any consecutive 2-Blocks with 4-Block. Remaining rows are constructed randomly.

Programming:

Interface for the visual model is written in JavaScrpit and works on any WebGL supported web browser. Graphics are built using WebGL framework THREE.JS (all libraries are included in the folder 'lib'). Application is made dynamic so that any size wall can be built. Orientation of wall can be changed using **q,w,e,a,s,d** keys. Script is in 'model.html'. Results of few test-cases are given below. You can always open '**model.html'** in browser to see the model for other dimensions.

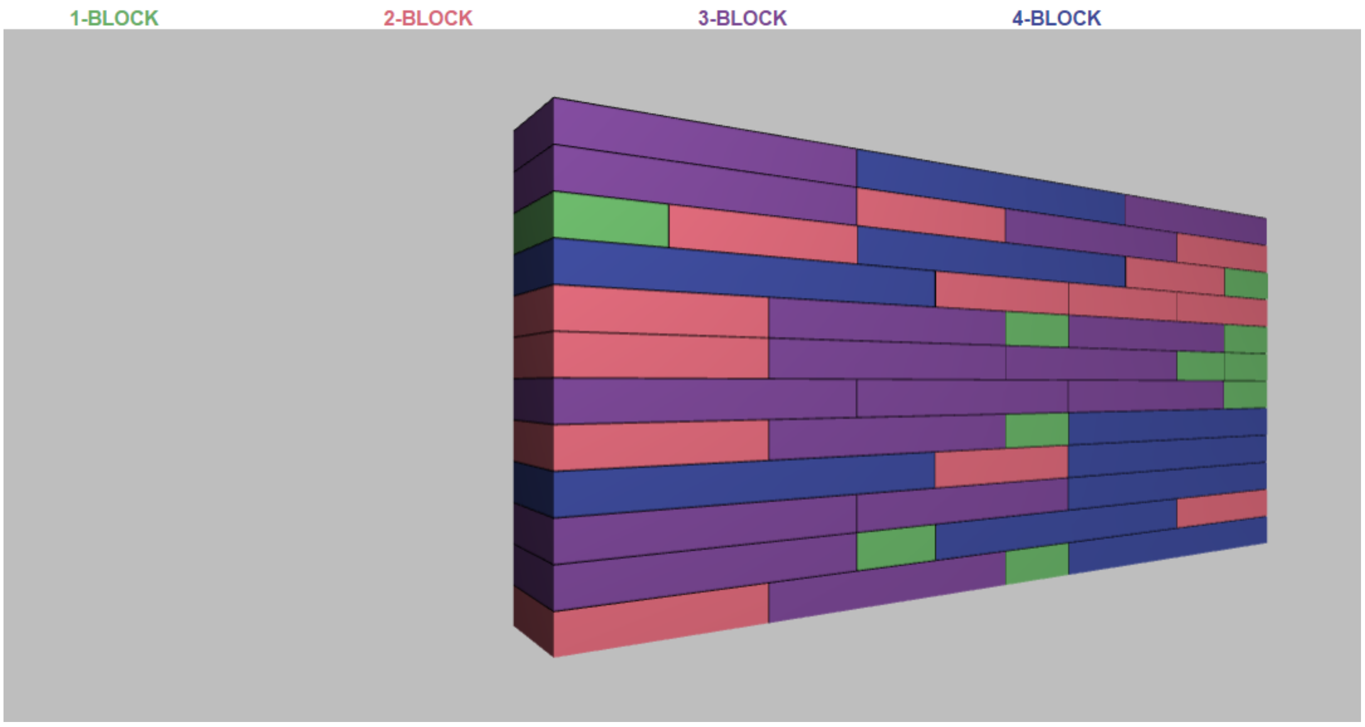
Test case 1: Wall Height = 10, Wall Width =12, No window



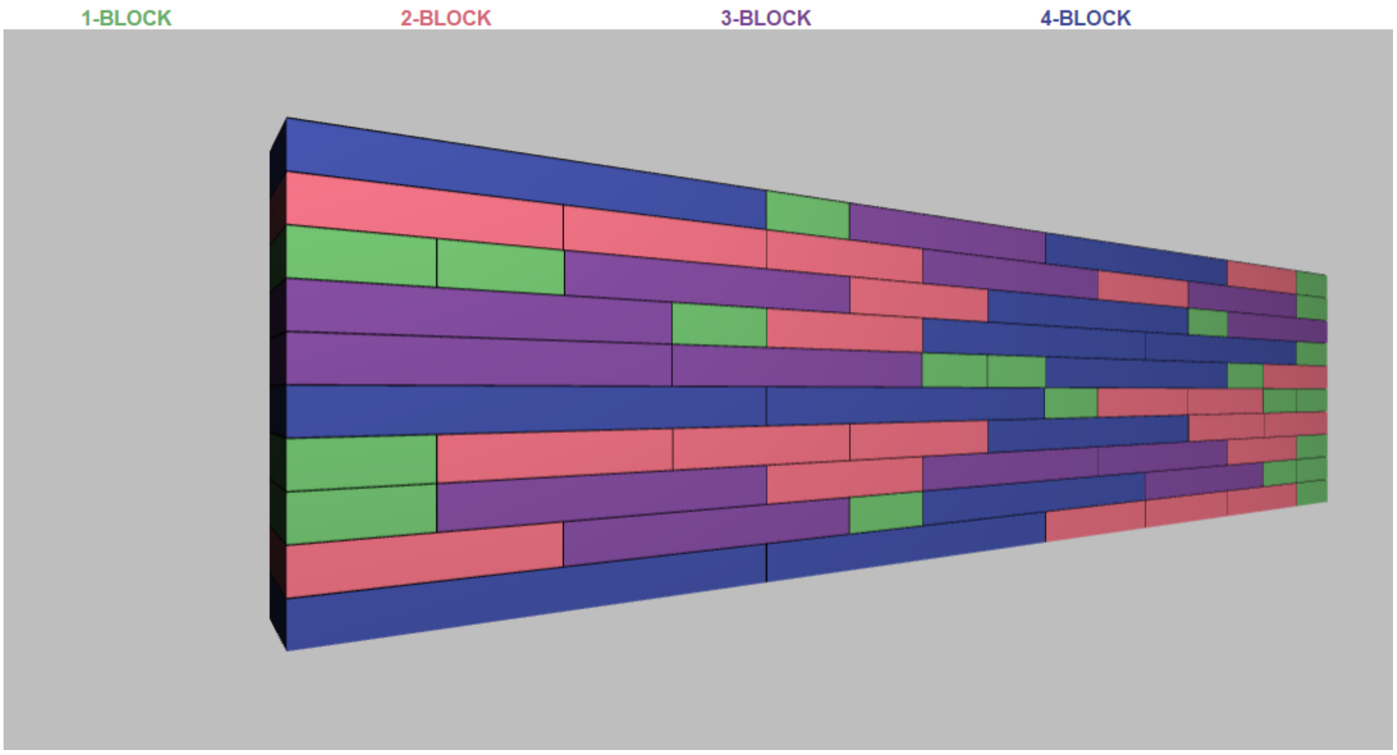
Visual model of the wall with/without window

Wall Height	Wall Width	Window Height	Window Width	Border
10	12	0	0	0.2

Test case 2: Wall Height = 12, Wall Width = 10, No window



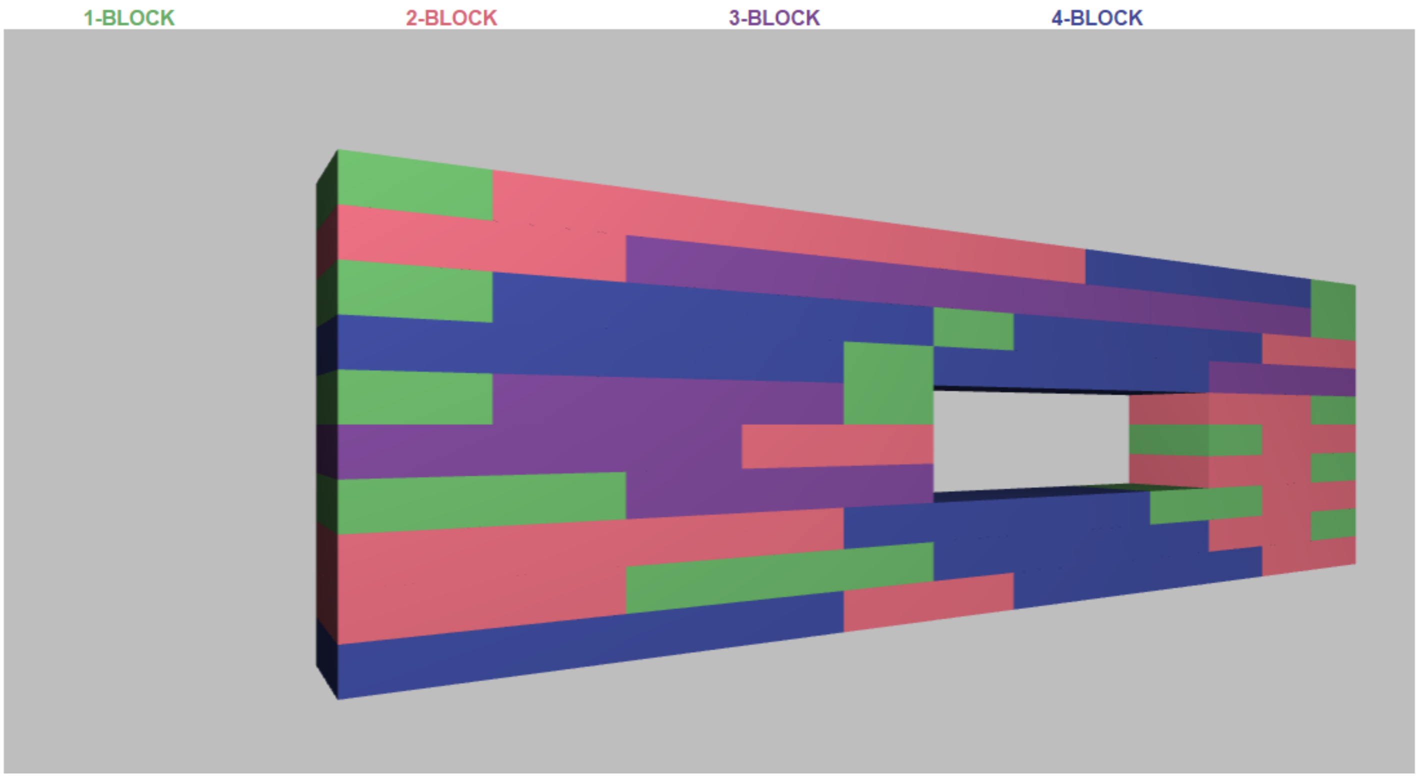
Test case 3: Wall Height = 12, Wall Width =15, No window



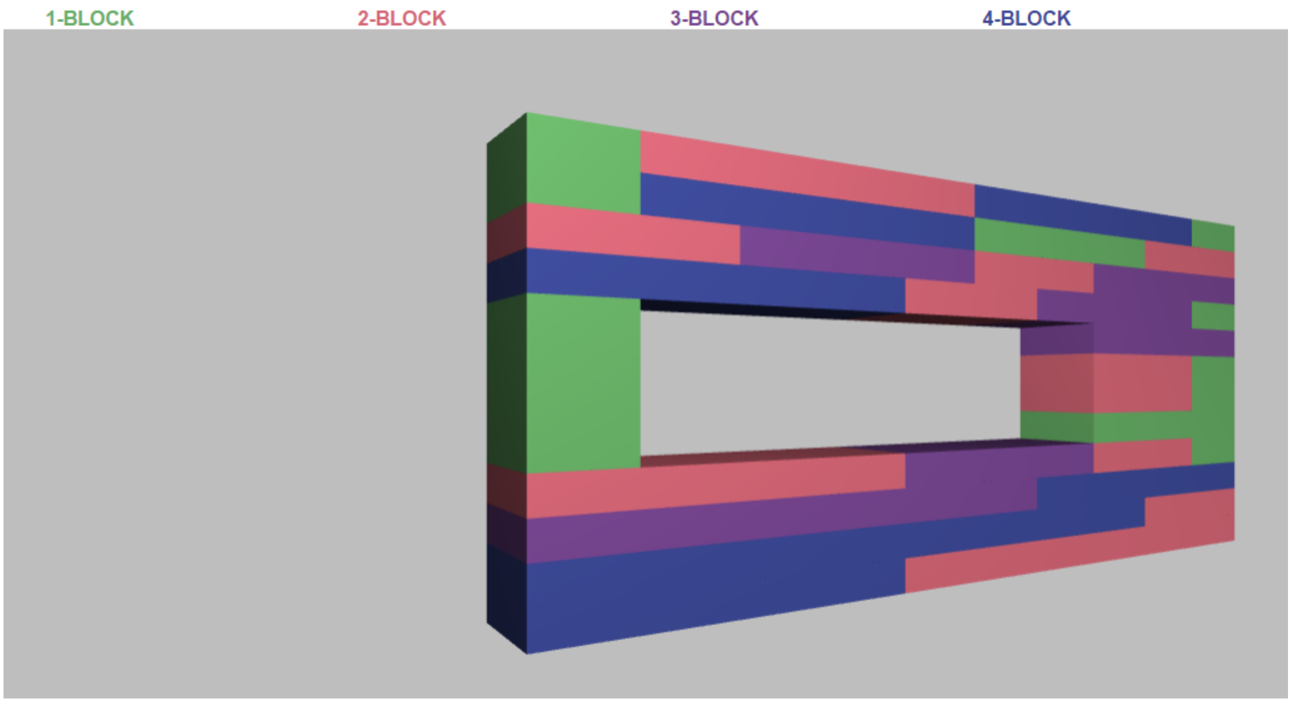
Ultimate Challenge: Build wall with Lego colors and windows

This is very much similar to the algorithm for wall without windows. Here also I ensure that there are at least 2 rows that are structurally connected (top and bottom rows are always connected). Feasible window location is found out randomly and after placing the window, remaining columns and rows are filled randomly with blocks. Results of few test cases are given below.

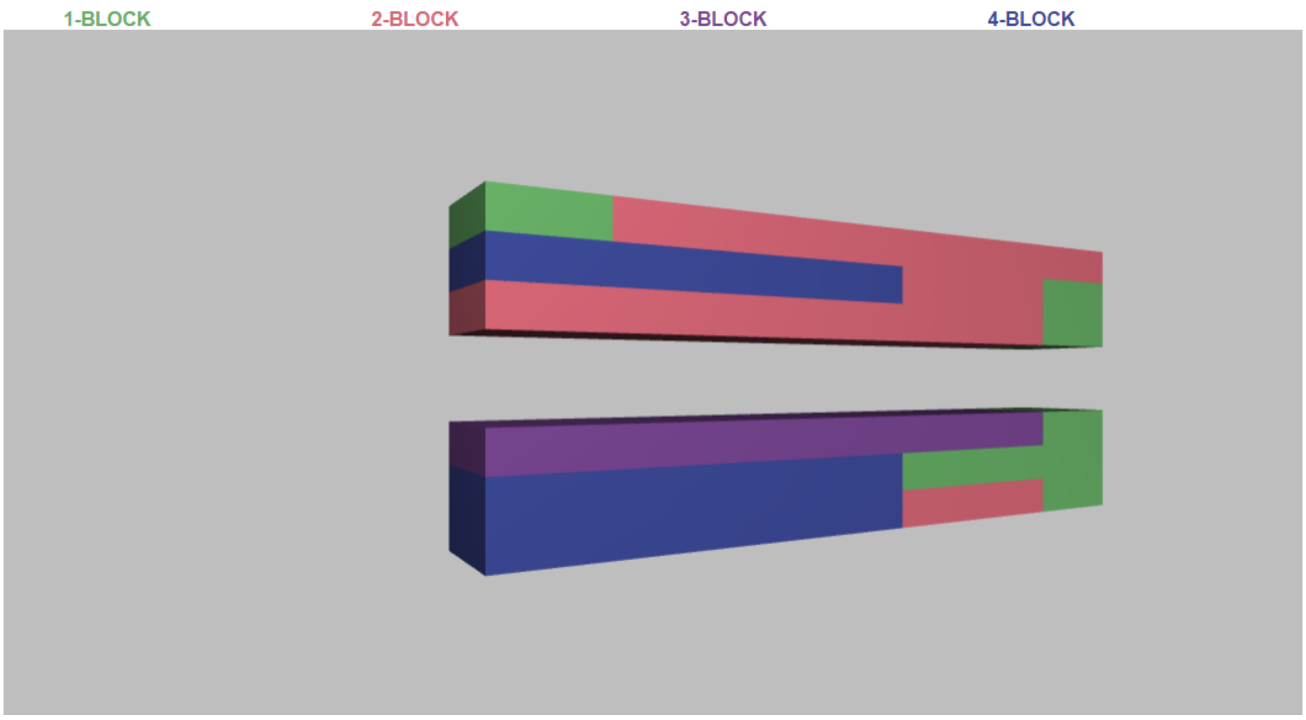
Test case 1: Wall Height = 10, Wall Width =12, Window Height = 3, Window Width = 4



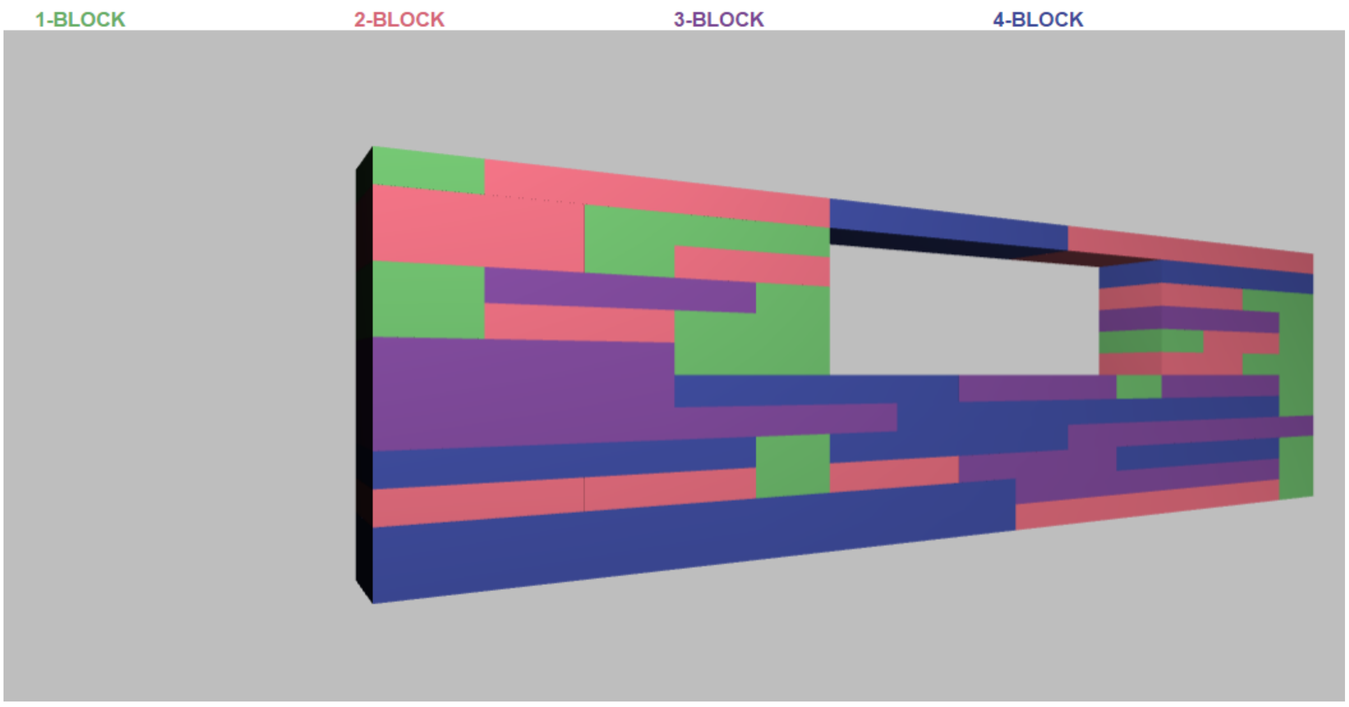
Test case 2: Wall Height = 12, Wall Width = 10, window height = 4, window width = 6



Test case 3: Wall Height = 8, Wall Width = 7, window height = 2, window width = 7 (extreme case)



Test case 4: Wall Height = 12, Wall Width =15, Window height =5, Width =6



Search Logs:

Minimum Challenge:
“Lego Block dynamic programming**”**
Medium and Ultimate Challenge:
I haven’t had much difficulty in coming up with the algorithm for building visual model, but I had to go through THREE.JS documentation to construct graphics.
Sites: “threejs.org”

Folder Structure:

solver.cpp – Finds number of ways without windows and prints it to stdout
model.html – Has the script for building visual model. Can use it by opening in a browser.
lib/ - Folder for the libraries used by model.html
Documentation – This file
solver.exe – binary for the solver.cpp