FIT5216: Modelling Discrete Optimization Problems
Assignment 1: Building Fences

1 Problem Statement

You are given a width W and height H rectangle of cells. Each cell has an associated cost to build a fence there, and an associated reward to build there (a negative reward indicates fences cannot be built there). You have a given budget which is the maximum spendable on the fence. You must determine the most profitable fence to build of (a) a given fixed length and (b) for any length up to that limit. The profit of the fence is the sum of the rewards minus the sum of its build cost. Each cell of the fence must be adjacent (horizontally or vertically) to the previous cell, and the whole fence must complete (the end is adjacent to the beginning).

Input data is given in MiniZinc data format:

```
W = ⟨ width ⟩;
H = ⟨ height ⟩;
cost = ⟨ 2D W × H array of costs ⟩;
reward = ⟨ 2D W × Harray of rewards ⟩;
length = ⟨ length (limit) ⟩;
budget = ⟨ build budget ⟩;
```

For example a sample data set is

```
W = 9;
H = 5;
cost = array2d(1..H,1..W,[1 | r in 1..H, c in 1..W]);
reward = [| 4, 0, 0, 0, 0, 0, 0, 0, 0
          | 0, 6, 0, 0, 2, 0, 0, 4, 0
          | 0, 0, 0, 0, 0, 0,-1, 0, 6
          | 0, 0, 4,-1, 0, 0, 0, 0, 0
          | 0, 0, 0, 0, 4, 0, 0, 2, 0
          |];
length = 12;
budget = 12;
```

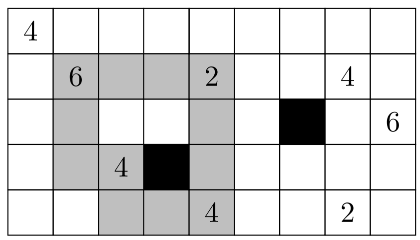
defines a problem on a 9x5 grid where the cost for each position is 1 and the reward data is given by

4								
	6			2			4	
						■		6
		4	■					
				4			2	

and the length (limit) is 12 and the budget 12. Note that in general the costs may be different for each cell, and the length different from the budget.

Part A - Fixed Length Fence

Create a model `fence.mzn` that takes data in the format specified above and decides on a fence of length equal to the value `length` given in the data.



Note that there are 12 cells in the fence, which is within the budget of 12, and the total reward is 16, although the cost is 12, hence the profit is 4. No cell is fenced with a negative reward (shown as filled in the diagram).

You model should define the positions of the cells as x and y coordinates, together with the total profit. One correct output for the solution above is

```
x = [2, 3, 4, 5, 5, 5, 5, 4, 3, 3, 2, 2];
y = [2, 2, 2, 2, 3, 4, 5, 5, 5, 4, 4, 3];
profit = 4;
```

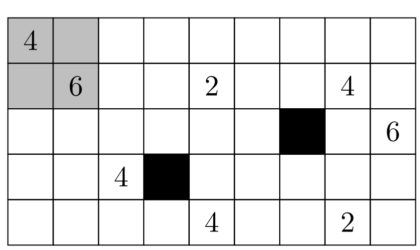
Note your output should have exactly this format, the value for `x`, `y` and `profit`.

Note how each position e.g. (2,2) is adjacent to the next one (3,2), including that the last position (2,3) is adjacent to the first (2,2). There are other correct outputs, starting at a different position, or going in the opposite direction.

Note that you will not be able to obtain full marks by just answering part A. Some problems will have no solution, whereas using part B they have a solution.

Part B - Bounded Length Fence

Modify your model `fence.mzn` to treat `length` as a bound on the maximal possible length of the fence. For example an optimal profit for the example data is illustrated by the solution



Note that there are 4 cells in the fence, which is within the budget and a total reward of 10, and a profit of 6.

Unused fence positions should be defined as having x and y coordinate 0. All the unused fence positions must occur at the end of the x and y lists. So a correct output for this solution is.

```
x = [2, 1, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0];
y = [2, 2, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0];
profit = 6;
```

2 Instructions

Edit the provided `mzn` model files to solve the problems described above. You are provided with some sample data files to try your model on. Your implementations can be tested locally by using the *Run* icon in the MiniZinc IDE or by using,

```
minizinc ./modelname.mzn ./datafile.dzn
```

at the command line.