ModRef 2022: Model and Solve Competition
Garden Paths

1 Problem Statement

You're a landscaper who has been asked to create a paved garden path. The garden is represented by a $W \times H$ area of hexagonal cells. The path starts in the leftmost column. Some cells feature a *reward*, because they contain some particularly nice flower beds or water features: If the path passes next to these cells, you pick up the reward. Some cells may have a *penalty* (negative reward), because that's where you e.g. place the compost bins or other things you don't want people to go near: If the path passes next to these cells, you are penalised accordingly. Your path cannot pass *through* any cell that has a positive or negative reward / penalty. You can only pick up the reward from a cell once, even if multiple cells of the path pass next to the cell.

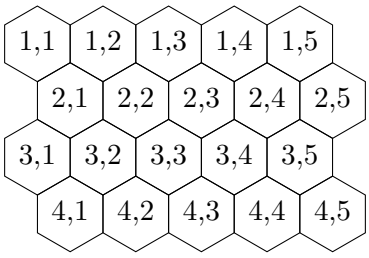
You are also given the maximum length (number of cells) of your path. Each cell of the path must be adjacent to the previous cell (i.e., the two cells must share an edge in the hexagonal grid). The path cannot visit a cell more than once. The path cannot have "branches".

The goal is to plan the path with the highest reward (sum of positive rewards and negative penalties) while using the shortest possible path. I.e., a shorter path with the same reward is considered better, while a shorter path with a lower reward is considered worse.

Input data is given in JSON data format:

```
{
"W" : ⟨ width ⟩,
"H" : ⟨ height ⟩,
"reward" : ⟨ 2D W × H array of rewards ⟩,
"length" : ⟨ length (limit) ⟩
}
```

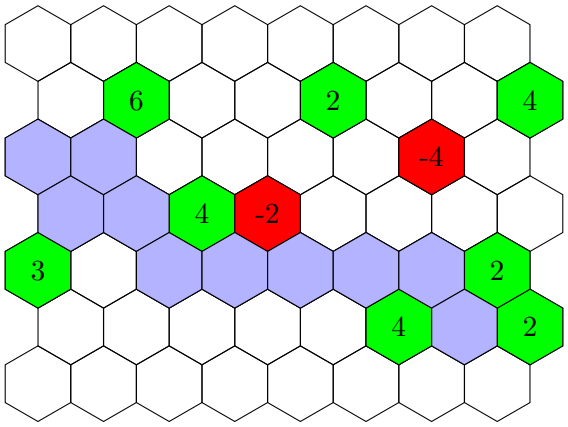
Here is an example of a grid with `W`=5 and `H`=4. The first row always starts to the left of the second row.



Here is a sample data set. Note that we have indicated the hexagonal grid in the reward array through indentation.

```
{
   "W" : 8,
   "H" : 7,
   "reward" : [ [0, 0, 0, 0, 0, 0, 0, 0],
                  [0, 6, 0, 0, 2, 0, 0, 4],
                [0, 0, 0, 0, 0, 0,-4, 0],
                  [0, 0, 4,-2, 0, 0, 0, 0],
                [3, 0, 0, 0, 0, 0, 0, 2],
                  [0, 0, 0, 0, 4, 0, 0, 2],
                [0, 0, 0, 0, 0, 0, 0, 0],
              ],
   "length" : 12
}
```

We can visualise the grid like this:



Green cells indicate positive rewards and red cells negative rewards (penalties). Light blue cells indicate a possible path. This path starts in the leftmost column as requied, has length 10 (within the limit of 12), and it achieves a total reward of $3 + 6 + 4 - 2 + 4 + 2 + 2 = 19$.

2 Output format

Your solutions must specify the path as the sequence of x and y coordinates of the cells. Note that x indicates the *column* and y indicates the *row*! The correct output for the solution above is

```
{
   "x" : [1, 1, 2, 2, 3, 4, 5, 6, 7, 7, 0, 0],
   "y" : [3, 4, 3, 4, 5, 5, 5, 5, 5, 6, 0, 0]
}
```

Since the path is shorter than the maximum length, fill the remainder of the x and y arrays with 0. I.e., your solution always has to contain `length` entries for x and y.