Please download `Challenge2.zip`. It contains project templates for the question, with a `main` and a header file containing the function prototype you should use.

Write a function with the following prototype:

```
void moveDL(int * block, int width, int height, int target)
```

This function should take in a pointer to a block of memory containing integers, which is `width` "wide" and `height` "tall". It should then find the first (starting from the top left corner and proceeding left to right and then top to bottom) instance of an integer equal to `target`. It should move this instance as far down and to the left as it can possibly go (whichever direction runs out of space first). The integer that was originally in the position the target now takes, should be placed where the target was originally. If it is not possible to shift the target, the function does not need to do anything. The block is altered "in place"; the function never needs to return anything whether it is successful or not. For instance, the following block

$$
\begin{array}{ccccc}
1 & 2 & \mathbf{0} & 4 & 5 \\
2 & 2 & 9 & 8 & 4 \\
\mathit{3} & 2 & 9 & 0 & 4 \\
6 & 6 & 6 & 6 & 6
\end{array}
$$

is called with

```
void moveDL(&block, 5, 4, 0)
```

Since the target is located at the top of the third column, it will need to move two rows down and two columns left, before it runs out of columns. This would leave the block looking like this:

$$
\begin{array}{ccccc}
1 & 2 & \mathit{3} & 4 & 5 \\
2 & 2 & 9 & 8 & 4 \\
\mathbf{0} & 2 & 9 & 0 & 4 \\
6 & 6 & 6 & 6 & 6
\end{array}
$$

Once the function has moved the first occurrence of `target`, it should end. It does not need to move subsequent occurrences, just the first one.