• State (in big-O notation) the implementation's memory complexity and briefly explain how you calculated this bound

```
this.data = (T[][]) new Object[width][height];
```

let's width be n, and height be m
memory complexity = O(n*m)

• Use this bound to evaluate the overall memory efficiency of your implementation - you should especially consider the case where your grid is very large but has very few elements

inefficient
Since, we need to go through whole array in order to find our element. If this array is very large, it will take lots of time to search. Besides, most of them are empty, but it will still occupy the memory, so it will waste lots of space to store null elements.

• Research and describe at least one alternative implementation that has significant advantages over the one chosen

HashMap <(x, y), T>
In this way, a hashmap uses keys and values, not indices. Therefore, you can only search for keys, and thus not access any index. So that it will be fast than 2d array. More important, it can save lots of space. We do not need to create memory for each element, if the element is null, we do not need to store it. In 2d array, once we initialise the array, it will automatically occupy memory for each index. But if we use HashMap, we can just store those elements which are not null (Stack Overflow, 2019).

• Compare the alternative implementation with your chosen approach in terms of both memory and runtime efficiency

HashMap:
Space Complexity of hashmap in big-O notation is O(n) where n is the number of entries (Anon, 2019).
Runtime Complexity: best and average case for Search, Insert and Delete is O(1) and worst case is O(n) where n is the number of entries (Skeet et al., 2019).

2d Array:
memory complexity = O(n^2) (if the size of column and row are n)
Runtime Complexity: for adding element is O(n), deleting element is O(n), get element is O(1), where n is the length of 2d array.

So clearly, no matter space complexity or runtime copmplexity, hashmap is a way better than 2d array, therefore, hashmap is more efficiency.

Reference:

Anon, (2019). [online] Available at: https://www.quora.com/What-is-the-Big-O-for-operations-in-a-Hashmap [Accessed 9 Aug. 2019].

Stack Overflow. (2019). *HashMap Space Complexity*. [online] Available at: https://stackoverflow.com/questions/43433699/hashmap-space-complexity?rq=1 [Accessed 9 Aug. 2019].

Skeet, J., Anderson, T., Ahle, T., Chalkias, K. and Verhas, P. (2019). *HashMap get/put complexity*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/4553624/hashmap-get-put-complexity [Accessed 9 Aug. 2019].