Looping 1 million times and summing elements in collections with 42 elements:

No optimization:

~2.80 millions of nanoseconds for the char array

~5.10 millions of nanoseconds for the 2D vector

~4.40 millions of nanoseconds for the string

O2 optimization:

Mainly in the 900 thousands of nanoseconds for the string.

Also averaging in the 900 thousands of nanoseconds (maybe slightly more variation?) for the 2D vector.

Seems to average on the upper half of the 900 thousands of nanoseconds for the char array. So a tad slower than the string and vector.

Conclusion:

* From this, it seems like there’s no significant difference in the collection used, if O2 optimization is turned on.
* If it’s turned off though, then the string > 2D vector, but the char array beats the string by even more.
* So for connect four, you could continue to turn the board into a string (you’ve already come this far), but it may not work out. However, using a proper hash table (instead of a huge 2D vector) could prove to be lucrative.
* <https://stackoverflow.com/questions/152745/optimising-c-2-d-arrays>
  + OP seems to have similar results as me. The answers give some ideas for optimization, when compiling with gcc (optimizations turned on).