When *N* is the number of items in the list, *n* is the index of the number to be calculated and *k* the current step to be calculated. And if *n >= ½N* (second half), the number to be calculated is just the sum of al previous numbers between *n*  and *N*.

So:

If you nest this trick you’ll get:

So this is going nowhere…

Just try something else. Suppose it ends with:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| start | Step 1 | Step 2 | Step 3 | Step 4 |
| T | T+….+Z | T + … + 7Z | T + … + 28Z | T + … + 84Z |
| U | U+V+W+X+Y+Z | U+2V+3W+4X+5Y+6Z | U+3V+… + 21Z | U+4V+10W+20X+35Y+56X |
| V | V+W+X+Y+Z | V+2W+3X+4Y+5Z | V+3W+6X+10Y+15Z | V+4W+10X+20Y+35Z |
| W | W+X+Y+Z | W+2X+3Y+4Z | W+3X+6Y+10Z | W+4X+10Y+20Z |
| X | X+Y+Z | X+2Y+3Z | X+3Y+6Z | X+4Y+10Z |
| Y | Y+Z | Y+2Z | Y+3Z | Y + 4Z |
| Z | Z | Z | Z | Z |

This is Pascal’s triangle, implementing this is also expensive. So forget it, I’m just going to do brute force.