**Recursive Strategy: Counting Zeroes**

|  |  |
| --- | --- |
| **Problem** | Count the zeroes in an array of ints, e.g. [ 0, 1, 1, 0, 1] Answer = 2 |
| **What is the smallest version of this problem?** *(leads to base case)* | If the array consists of one element, e.g. [ 0 ] or [ 1 ], return 1 if the element is a zero, else return 0. |
| **What recursion strategy should I use:**   * **Forwards recursion** *(each recursive step gets larger; the base case is based on  # of iterations)* * **Backwards recursion** 🡨 classical *(each recursive step gets smaller until the base case is reached)* | Backwards recursion |
| **For recursive cases, should I**:   * **Process first and recur last?** *(process as I move up the recursive stack)* * **Recur first and process last?** *(process as I move down the recursive stack)* | Process as I move **up** the stack. |
| **What should each recursive step do?** | Keep a running total of the number of zeroes seen.  If the first element of the array is a 0, increment running total; i.e. keep sending the recursive function an increasingly large count of zeroes. |
| **For backwards recursion solutions:**  **How should the problem be reduced on each step?** | Send progressively smaller Arrays on each iteration where each Array is missing its first element.\* |
| **For forwards recursion solutions:**  **How should I keep track of the running answer?** | N/A |

\*Instead of sending smaller and smaller arrays each time, could you also send the ***same array*** to each step as well as the starting position for evaluation (which would increment by one on each recursion)?