L2: Elementary Problem

# How to identify if the problem can be solved with Dynamic Programming

* There are two sub classes of dynamic programming problems, mentioned below:
  + **Combinatoric:** They answer the question***“How Many …?”*.** The end goal in such problems is to count something**.** Some common types of these questions are:
    - ***How many ways to make a change?***
    - ***How many ways to traverse a graph?***
    - ***How many steps needed to get from point A to B?***
  + **Optimization:** In these problems we need to find a strategy that maximizes or minimizes some function. Some common types of these questions are:
    - ***What is the maximum profit gained by buying and selling a stock?***
    - ***What is the minimum cost to travel from New York to Mumbai?***
* Dynamic programming relies on the fact that the optimal solution to the optimization problems or combinatorial problem depends on the optimal solution for its overlapping sub problems.

# Problem 1: Find the sum of the first N numbers

* Break the problem into multiple subproblem, the simplest possible subproblem is for N=1 in this case the smallest possible sum is 1. Hence f(N=1) = 1.
* Let’s progress to N=2**,** now we must calculate f(N=2). This problem can be written as f(N=1) + 2. This is how we devise a general formula for the elements.
* Hence the general problem for our problem statement is: f(N) = f(N-1) + N
* Python Code available [here](https://github.com/Mcdaddy-2pointoh/Dynamic-Programming/blob/main/Coded%20solutions/NSum(Problem1).ipynb).

Initial Problem

Subproblem Fragmentation