**Divide and conquer**

**Divide and conquer** is an algorithm design paradigm in which a problem is divided into smaller subproblems (often two ones) of the same type and then each subproblem is solved independently. The division is applied recursively until sub-problems become simple enough to be solved directly using a base case. Finally, the solutions of all sub-problems are combined to get the solution for the original problem. Let's consider each of the described steps in more detail.

## The steps of a divide and conquer based algorithm

A typical algorithm based on the divide and conquer paradigm consists of three steps:

1. **Divide:** split a problem into smaller sub-problems of the same type. Each sub-problem should represent a part of the original problem.
2. **Conquer:**recursively solve the sub-problems. If they are simple enough, solve them directly using base case conditions.
3. **Combine:** unite the solutions of the sub-problems to get the solution for the original problem.

## A simple example: the sum of elements in an array

Let's consider how the divide and conquer paradigm can be used to calculate the sum of elements in an array. Note that the problem can be solved more efficiently and in a more simple way. Here, we apply the paradigm just to get a better understanding of how it works. The procedure is the following:

calc\_sum(array, left, right):  
    # the sum of zero elements is 0  
    if left == right:  
        return 0      
   
    # the sum of one-element sub-array is the element  
    if left == right - 1:  
        return array[left]  
   
    # the index of the middle element to divide the array into two sub-arrays  
    middle = (left + right) / 2;  
   
    # the sum of elements in the left subarray  
    left\_sum = calc\_sum(array, left, middle)  
   
    # the sum of elements in the right subarray  
    right\_sum = calc\_sum(array, middle, right)  
   
    # the sum of elements in the array  
    return left\_sum + right\_sum

single\_elem\_arr = [55]  
sum1 = calc\_sum(single\_elem\_arr, 0, 1) # 55  
   
two\_elems\_arr = [14, 36]  
sum2 = calc\_sum(two\_elems\_array, 0, 2) # 50  
   
five\_elems\_arr = [14, 27, 31, 54, 38]  
sum3 = calc\_sum(five\_elems\_arr, 0, 5) # 164