Divide and conquer is a recursion-based algorithm design technique in which a problem is broken down into smaller sub-problems, and then solutions to sub-problems are combined to come up with a solution to the main problem. Problems are broken down to smaller sub-problems until these sub-problems are so simple that they can be solved directly (usually in a constant time).

General plan for divide and conquer algorithms:

1. An instance of the problem is divided into a number of smaller instances of the problem.
2. The smaller instances of the problem are solved recursively (make sure you have an appropriate stop condition).
3. Solutions of smaller instances are combined to get a solution to the original instance.

### 5.1 Recursion

A recursive algorithm is simply one that calls itself. A recursive algorithm must have an appropriate stop condition that prevents it from calling itself forever!

Example:
Consider an algorithm that computes the factorial of a positive integer.

Example:
Consider an algorithm that finds the sum of an array of \( n \) integer numbers.

---

5. Divide and conquer

Algorithm 1: A recursive algorithm that computes the factorial of a positive integer.

**Algorithm Factorial**$(n)$

- if $n = 0$ then
  - return 1;
- else
  - return $\text{Factorial}(n - 1) \ast n$;

Algorithm 2: Finding the sum of $n$ integers.

**Algorithm Sum**$(A[], first, last)$

- if $first = last$ then
  - return $A[first]$;
- else
  - return $\text{Sum}(A[], first, first + \left\lceil \frac{last - first}{2} \right\rceil - 1) +$
  - \hspace{1cm} $\text{Sum}(A[], first + \left\lceil \frac{last - first}{2} \right\rceil, last)$;

% First call to this algorithm should be Sum $(A[], 0, n - 1)$;

Example:

Tower of Hanoi: “It consists of three rods, and a number of disks of different sizes which can slide onto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on one rod, the smallest at the top, thus making a conical shape. The objective of the puzzle is to move the entire stack to another rod, obeying the following rules:

- Only one disk may be moved at a time.
- Each move consists of taking the upper disk from one of the rods and sliding it onto another rod, on top of the other disks that may already be present on that rod.
- No disk may be placed on top of a smaller disk.”

\[^2\text{From Wikipedia.}\]
Algorithm 3: Tower of Hanoi (moving $n$ disks from rod $x$ to rod $z$ through rod $y$).

Algorithm TOH(int $n$, char $x$, char $y$, char $z$)
if $n = 1$
    Println (“move the top disk from rod ” $x$ “to the top of rod ” $z$);
else
    TOH($n - 1$, $x$, $y$, $z$);
    TOH(1, $x$, $y$, $z$);
    TOH($n - 1$, $y$, $x$, $z$);
end