Exhaustive search

- **exhaustive search**: Exploring every possible combination from a set of choices or values.
  - often implemented recursively

  Applications:
  - producing all permutations of a set of values
  - enumerating all possible names, passwords, etc.
  - combinatorics and logic programming

- Often the search space consists of many ***decisions***, each of which has several available ***choices***.
  - Example: When enumerating all 5-letter strings, each of the 5 letters is a *decision*, and each of those decisions has 26 possible *choices*.

Exhaustive search

A general pseudo-code algorithm for exhaustive search:

```
function Search (decisions):
    – If there are decisions left to make:
        // Let's handle one decision ourselves, and the rest by recursion.
        • For each available choice C for this decision:
            – Choose C.
            – Search the remaining decisions that could follow C.
    – Otherwise, if there are no more decisions to make:  Stop.
```

– *Observation:* The "**base case**" no longer represents a simple case of the algorithm; rather, it is the case where the algorithm is finished working and has no more choices left to make.