



Subsets: Introduction

Let's understand the Subsets pattern, its real-world applications, and some problems we can solve with it.

We'll cover the following



- Overview
- Example
- Does my problem match this pattern?
- Real-world problems
- Strategy time!

Overview

The **subsets** pattern is useful in finding the permutations and combinations of elements in a data structure. The input data structure might be a set containing unique elements. It can also be an array, or a list, which may contain duplicate elements. We then make a series of subsets from the elements in this data structure. The specific subsets generated are based on the conditions that the problem provides us.

We use backtracking to generate the required subsets of a given data structure of elements. The method is to build the subsets incrementally, including or excluding each element of the original data structure, depending on the constraints of the problem. This process is continued recursively for the remaining elements until all desired subsets have been generated.

The following illustration shows how subsets are made from a given array:



We start with an empty set, iterate through all elements, and add them to all existing sets to create subsets.

array	1	2	3
-------	---	---	---

{}

1 of 4



Note: While subsets can be generated using backtracking, not all problems solved using backtracking involve subsets. Backtracking applies to a broader range of problems where exhaustive search, that is, evaluating all possibilities, is required. These problems may involve various constraints, rules, or conditions that guide the exploration process.

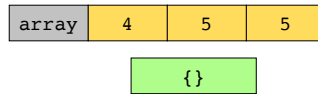
Example

The following example illustrates a problem that can be solved with this approach:



Find all unique subsets in an array that may contain duplicate elements

Sort the array first, then traverse each element of the array. If we haven't seen the current element, we add it to all the existing subsets to make new subsets. Otherwise, we add the element only to the subsets of the previous iteration to make new subsets.



1 of 6



Does my problem match this pattern?

- Yes, if the following condition is fulfilled:
 - The problem requires creating permutations or combinations of the elements in the input data structure, either as the final solution or as an intermediate step.

Additionally, the problem may specify a condition for including a particular permutation or combination in the final solution.

- No, if the following condition is fulfilled:
 - We are not required to generate permutations or combinations of the elements in the input data structure.

Real-world problems

Many problems in the real world use the subsets pattern. Let's look at some examples.

- **Movie combinations:** Create a combination of movies, chosen from the given list of genres in a specific sequence.

- **Movie viewing orders:** Generate all possible permutations of a given list of movies to provide options for a movie marathon.
- **Total cost of shopping items:** An equation calculating the total cost



Strategy time!

Match the problems that can be solved using the subsets pattern.

Note: Select a problem in the left-hand column by clicking it, and then click one of the two options in the right-hand column.

Match The Answer

ⓘ Select an option from the left-hand side

Find the longest possible route of 1s in a matrix of 0s and 1s

Find any one subset in a set of elements whose sum adds up to 12

Generate the power set of the set $\{1, 2, 3\}$

Shift all 0s in an array to

Subsets

Some other pattern



the right

Reset

Show Solution

Submit



