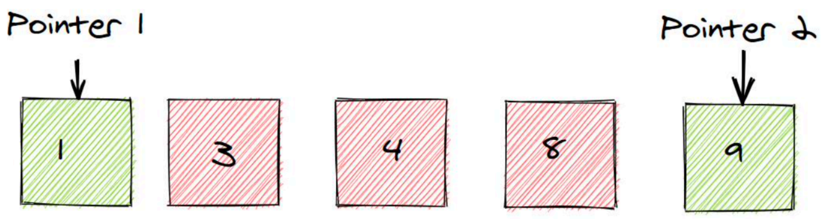
Two Pointer Technique

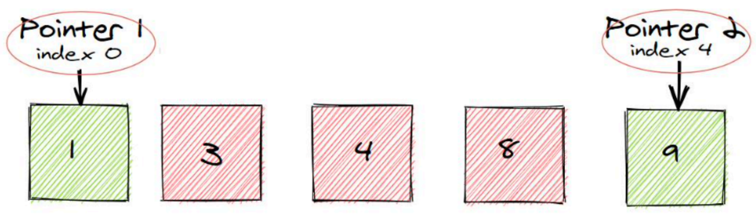
The **two pointer technique** is a near necessity in any software developer's toolkit, especially when it comes to technical interviews. In this guide, we'll cover the basics so that you know when and how to use this technique.

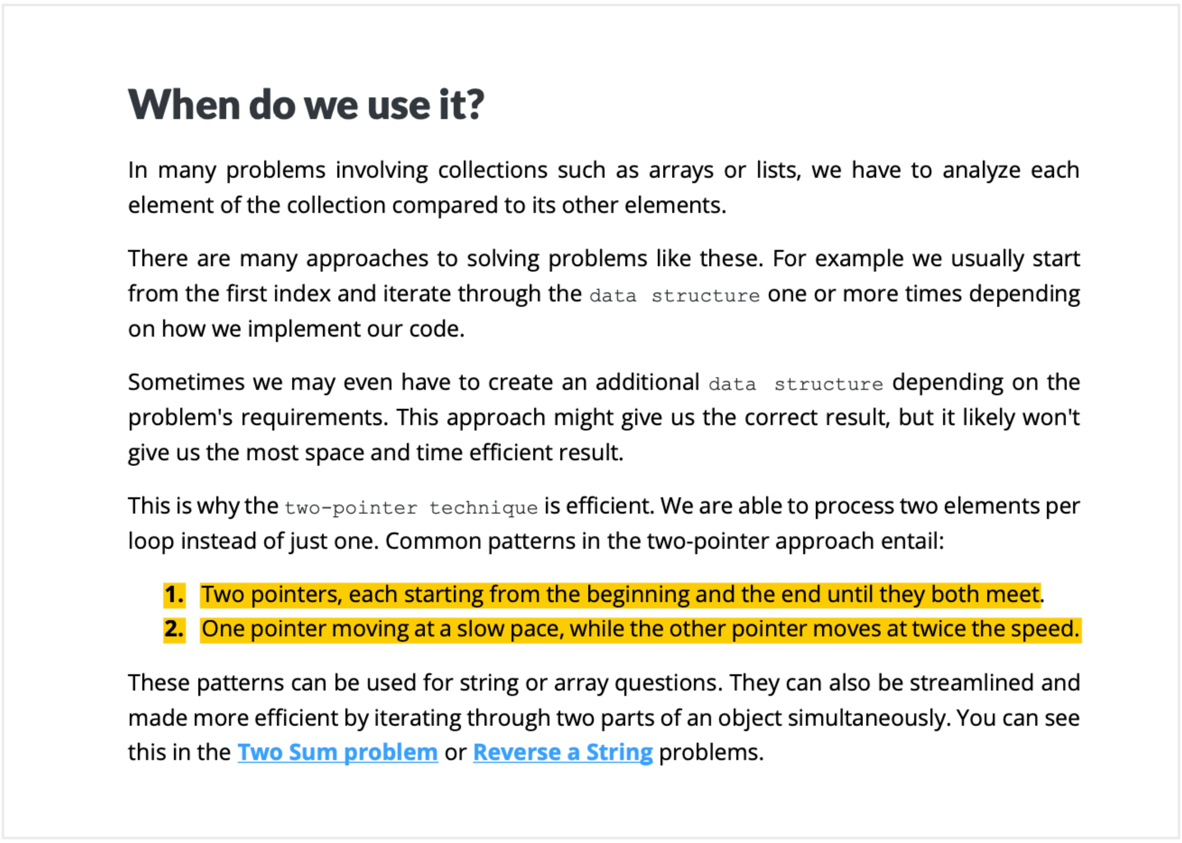


What is the pattern?

The name `two pointers` does justice in this case, as it is exactly as it sounds. It's the use of two different pointers (usually to keep track of array or string indices) to solve a problem involving said indices with the benefit of saving time and space. See the below for the two pointers highlighted in yellow.

But what are `pointers`? In computer science, a `pointer` is a reference to an object. In many programming languages, that object stores a memory address of another value located in computer memory, or in some cases, that of memory-mapped computer hardware.





When do we use it?

In many problems involving collections such as arrays or lists, we have to analyze each element of the collection compared to its other elements.

There are many approaches to solving problems like these. For example we usually start from the first index and iterate through the `data structure` one or more times depending on how we implement our code.

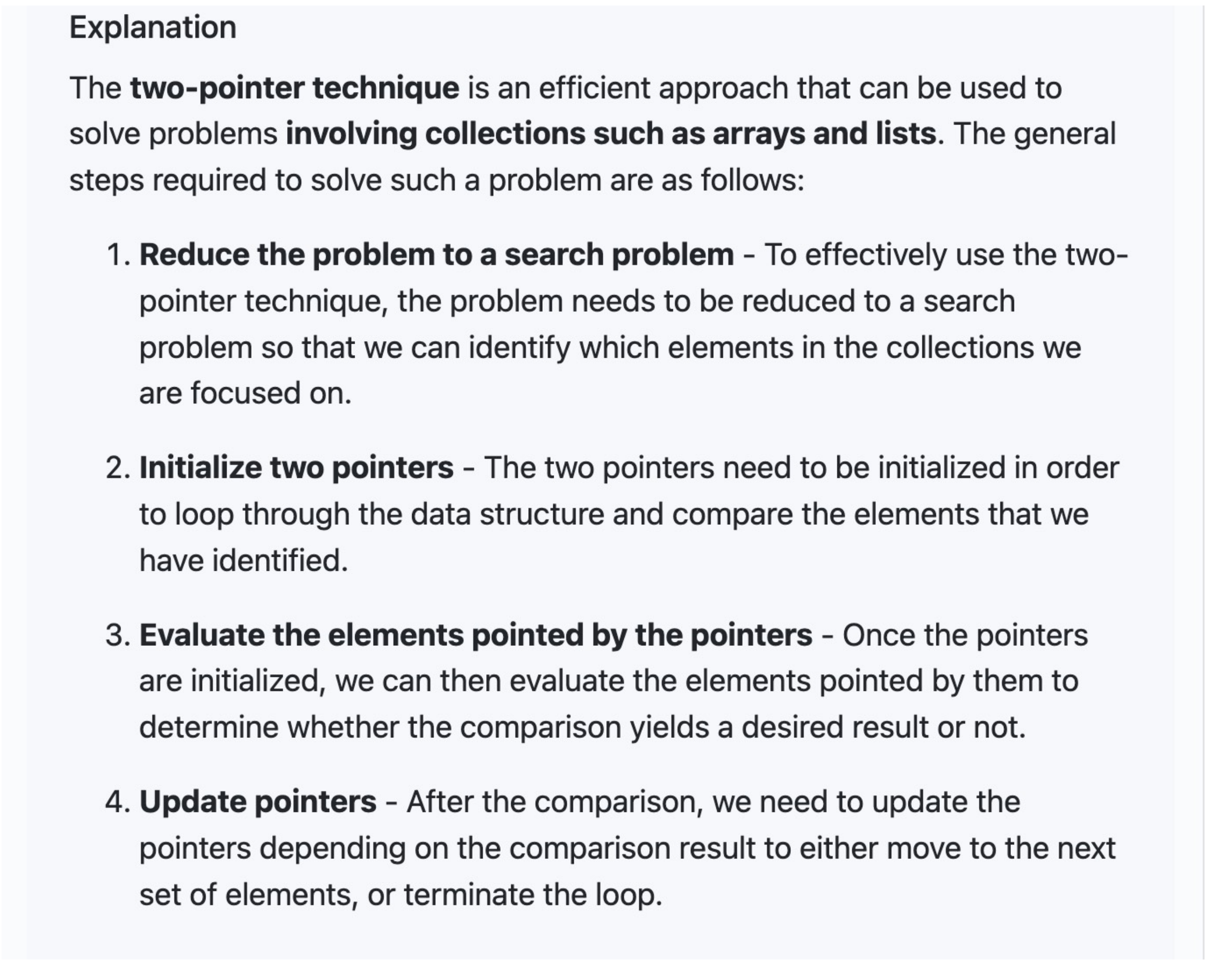
Sometimes we may even have to create an additional `data structure` depending on the problem's requirements. This approach might give us the correct result, but it likely won't give us the most space and time efficient result.

This is why the `two-pointer technique` is efficient. We are able to process two elements per loop instead of just one. Common patterns in the two-pointer approach entail:

1. Two pointers, each starting from the beginning and the end until they both meet.
2. One pointer moving at a slow pace, while the other pointer moves at twice the speed.

These patterns can be used for string or array questions. They can also be streamlined and made more efficient by iterating through two parts of an object simultaneously. You can see this in the **Two Sum problem** or **Reverse a String** problems.

This technique can be used for strings , arrays and some linked-list questions.



Explanation

The **two-pointer technique** is an efficient approach that can be used to solve problems **involving collections such as arrays and lists**. The general steps required to solve such a problem are as follows:

1. **Reduce the problem to a search problem** - To effectively use the two-pointer technique, the problem needs to be reduced to a search problem so that we can identify which elements in the collections we are focused on.

2. **Initialize two pointers** - The two pointers need to be initialized in order to loop through the data structure and compare the elements that we have identified.

3. **Evaluate the elements pointed by the pointers** - Once the pointers are initialized, we can then evaluate the elements pointed by them to determine whether the comparison yields a desired result or not.

4. **Update pointers** - After the comparison, we need to update the pointers depending on the comparison result to either move to the next set of elements, or terminate the loop.