Arrays

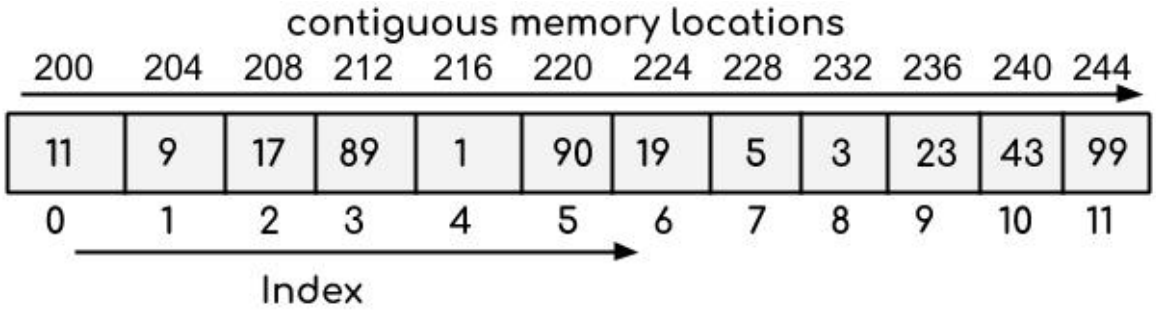
Learn about the array data structure, how it differs from lists and sets, and how you can use them in Kotlin.

WE'LL COVER THE FOLLOWING

Like lists and sets, arrays store multiple values in a linear data structure.

What is an Array?

An array is a basic data structure in all programming languages. It stores multiple elements by placing them consecutively in memory. This structure is represented in the following illustration:



Array data structure. Elements are stored contiguously in memory and index starts at zero. [Source: https://beginnersbook.com/2018/10/data-structure-array/]

What's the Difference Between Arrays, Lists, and Sets?

Sets are easy to distinguish from both lists and arrays based on their special properties: no ordering and no duplicates.

To understand the difference between lists and arrays, we'll have to look at the way their elements are stored and what implications this has for us as developers.

While arrays store their values in one consecutive part of computer memory, list and set elements may be spread across memory. This has several implications:

- **Arrays are a fixed-length data structure.** This is because, after the array is initially created, the next consecutive place in memory may be taken by another value. Thus, no more consecutive values can be added to the array. Not being able to add or remove elements distinguishes them from *mutable* lists.
- Differences between arrays and read-only lists exists only on a lower level, i.e., in the exact API they provide, the storage structure in memory, and their resulting runtime performance.

Finally, there's no differentiation between read-only and mutable arrays in Kotlin. Array elements can be overwritten but no elements can be added or removed, as they're fixed in length.

> **Note:** Recall that arrays are not part of Kotlin's Collections API. But they have a similar data structure, so they're mentioned here. All actual *collections* have read-only and mutable variants.

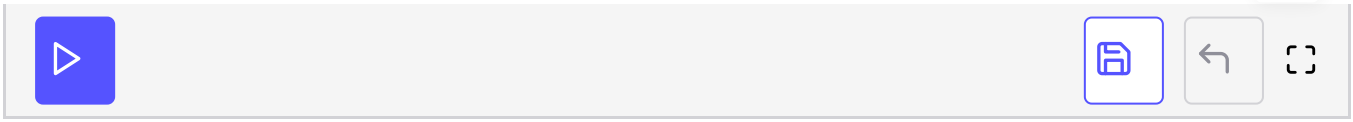
Data structure	Read-only available?	Ordered?	Duplicates allowed?	Memory storage
List	Yes	Yes	Yes	Non-sequential
Set	Yes	No	No	Non-

Set	Yes	No	No	sequential
Array	No	Yes	Yes	Sequential

Creating an Array

Kotlin's API is very consistent when it comes to helper functions so the way to create an array should look familiar:

```
val priorities = arrayOf("HIGH", "MEDIUM", "LOW")
```

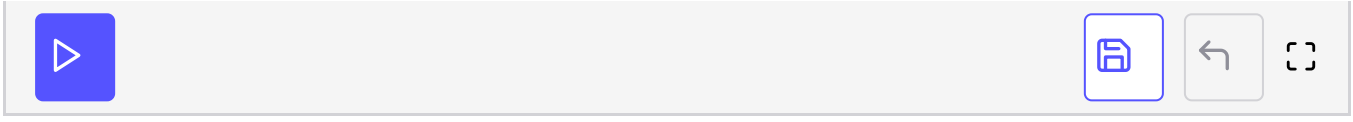


The `arrayOf` function accepts any number of arguments and returns an `Array<Something>`. In the example, it returns an `Array<String>` (reads "array of string").

Working with Arrays

While you can't add or remove elements from an array, you *can* access and manipulate all its elements using the indexed access operator:

```
val priorities = arrayOf("HIGH", "MEDIUM", "LOW")
println(priorities[1])
priorities[1] = "NORMAL"
println(priorities[1])
println(priorities.size)
```



As you can see, the way to access and manipulate existing elements is consistent across all linear data structures.

Quiz

Arrays in Kotlin

1

Which of the following are properties that make *arrays* different from *sets*? You can select multiple answers.

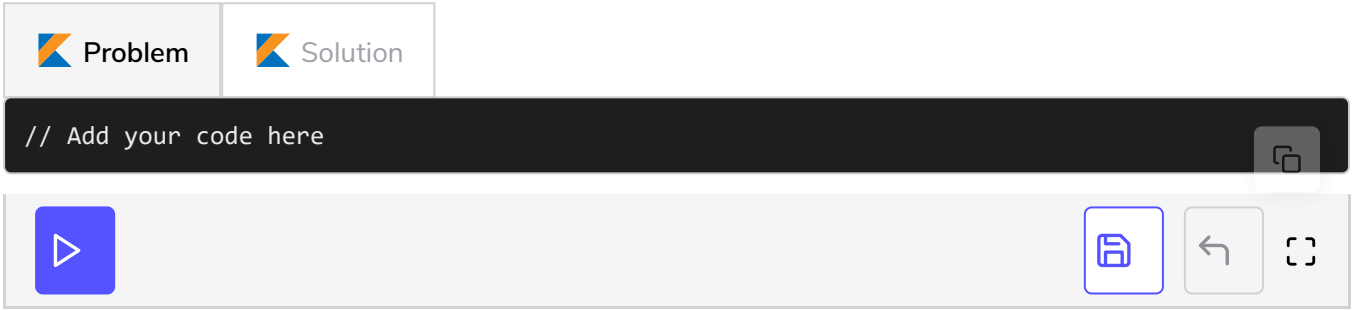
COMPLETED 0%

1 of 3

Exercises

Initialize an array of `programmingLanguages` you're familiar with.

- Print the first element of your array to the console
- Overwrite the first element by the next programming language you want to learn
- Print the first element of your array to the console again to verify it was changed



Summary

Arrays are a linear data structure that store multiple values in memory sequentially.

- In contrast to sets, arrays are ordered, may contain duplicates, and store

elements sequentially in memory.

- In contrast to lists, arrays are fixed-length because they store elements sequentially, and are always mutable.
- You can create arrays using `arrayOf(...)`.

The next lesson introduces maps, the first non-linear collection data structure.