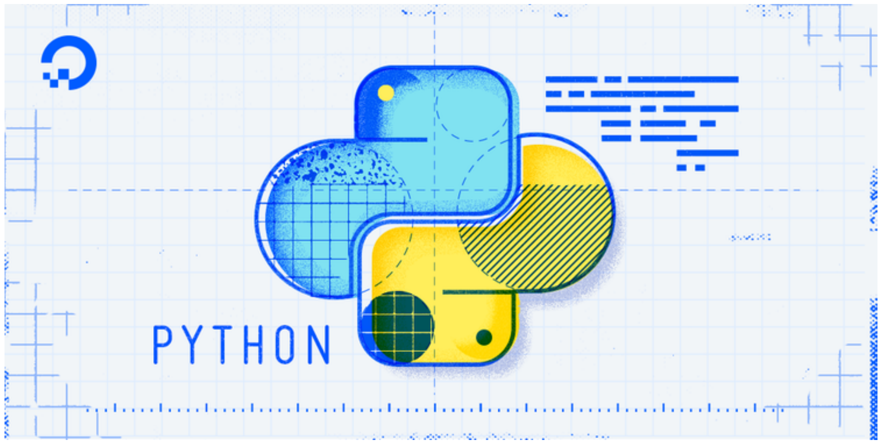
Bolaji Follow
Tech Enthusiast. Software Dev. at @Andela
May 5 · 3 min read



List vs Array — Data Types

Some days back, I was working on a Python project and I had to make use of lists. I created an empty list and named it `cost_array` . A friend saw this and asked the question:

> *"What is the difference between a List and an Array?"*

I began to ponder because before now I've always regarded data of type `['Bolaji', 'Proton']` as lists in Python and arrays in Javascript. Apparently, an Array is a data type in Python also, meaning we have the `array` type and `list` type (the list type being more popular). Most people get to use arrays when they venture into Data Science and make use of libraries such as `numpy` . Arrays are the real workhorse of data structures for scientific and engineering applications. The most popular type of array used in Python is the `numpy` array.

Similarities between Lists and Arrays

- Both are used for storing data
- Both are mutable
- Both can be indexed and iterated through
- Both can be sliced

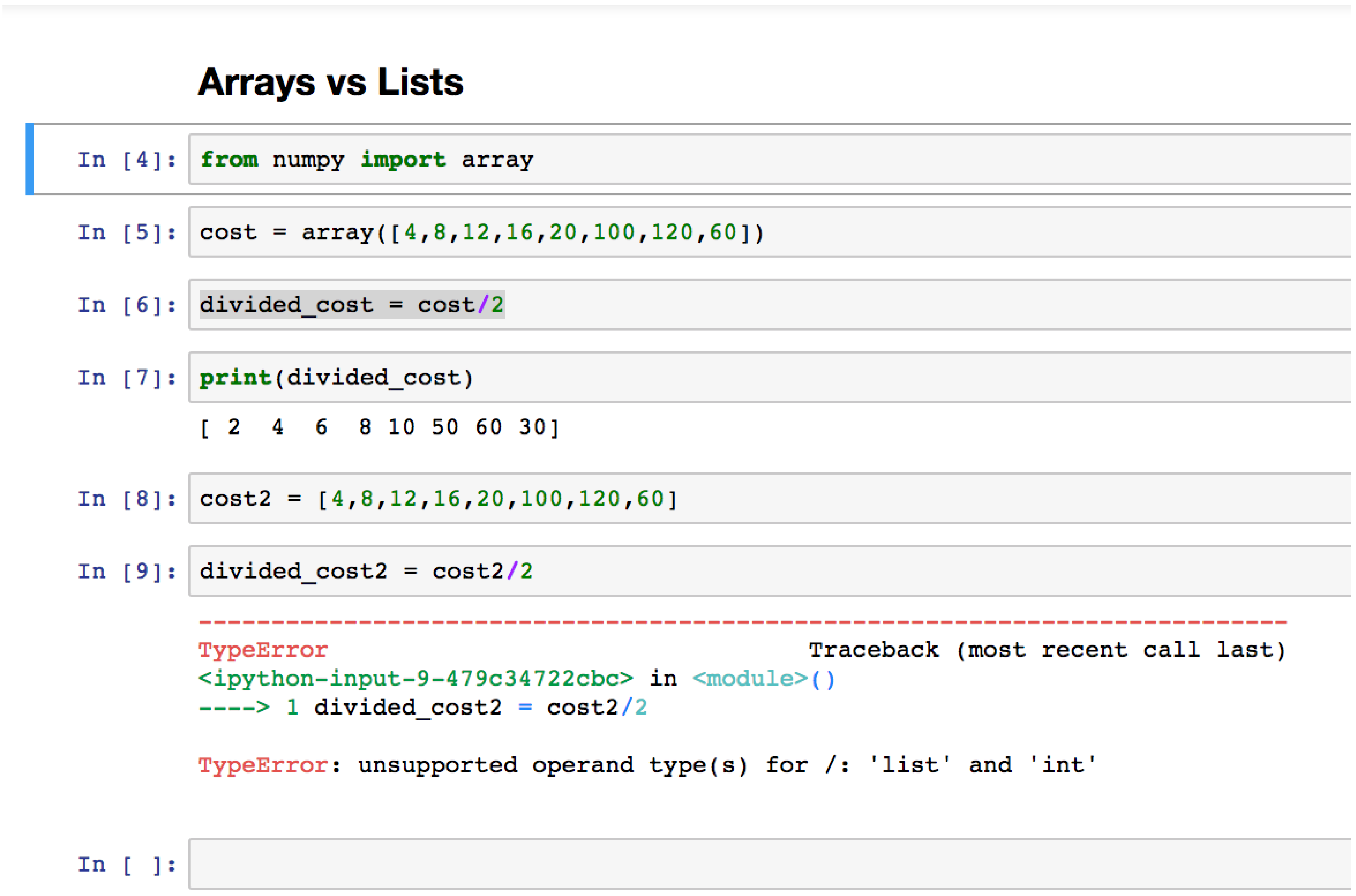
Differences

The main difference between these two data types is the operation you can perform on them. Arrays are specially optimised for arithmetic computations so if you're going to perform similar operations you should consider using an array instead of a list.

Also lists are containers for elements having differing data types but arrays are used as containers for elements of the same data type.

The example below is the result of dividing an array by a certain number and doing the same for a list. When we try the same operation (example: division) on a list, we get a TypeError because builtin python lists do not support the `__div__` protocol. It takes an extra step to perform this calculation on a list because then you'd have to loop over each item one after the other and save to another list.

Stress!!



Arrays vs Lists

```
In [4]: from numpy import array

In [5]: cost = array([4,8,12,16,20,100,120,60])

In [6]: divided_cost = cost/2

In [7]: print(divided_cost)

        [  2   4   6   8  10  50  60  30]

In [8]: cost2 = [4,8,12,16,20,100,120,60]

In [9]: divided_cost2 = cost2/2

        ---------------------------------------------------------------------------
        TypeError                                 Traceback (most recent call last)
        <ipython-input-9-479c34722cbc> in <module>()
        ----> 1 divided_cost2 = cost2/2

        TypeError: unsupported operand type(s) for /: 'list' and 'int'

In [ ]:
```

Making use of more robust array data types isn't also without it's cost implications. For example, to use `numpy` arrays you need to introduce a dependency in your project on the `numpy` library. You'd have to install the `numpy` package, import it and declare it while a list can be created on the fly.

While `numpy` provides a more robust array data type suited for numerical computation, Python also has a builtin array data type.

Before you go…

Thanks for reading!

If you enjoyed this article, please hold down the clap button □ to help others find it. The longer you hold it, the more claps you give!

And do not hesitate to share your thoughts in the comments below.