Pandas Core Components - The Series Object

WE'LL COVER THE FOLLOWING

1. Series From Lists and Arrays

A Pandas Series is a *1D* array of indexed data essentially a column. It can be created from a list or an array using the `pd.Series()` method as shown in the code-widget below.

Run the code and **observe** the output to understand the concepts in a hands-on example.

Notice that the standard shorthand for importing Pandas is *pd*.

```
import pandas as pd
series = pd.Series([0, 1, 2, 3])
print(series)
```

From the previous output, we can see that a Series consists of both a sequence of values and a sequence of indices. The values are simply a NumPy array, while the index is an array-like object of type `pd.Index`. Values can be accessed with the corresponding index using the already familiar **square-bracket and slicing notations**:

```
import pandas as pd
series = pd.Series([0, 1, 2, 3, 4, 5])

print("Values:", series.values)
```

```
print("Indices:", series.index, "\n")

print(series[1], "\n")   # Get a single value


print(series[1:4]) # Get a range of values
```

But why should we use Series when we have NumPy arrays?

Pandas' Series are much more general and flexible than the *1D* NumPy arrays. The essential difference is the presence of the index; while the values in the **NumPy array have an implicitly defined integer index** (to get and set values), the **Pandas Series has an explicitly defined integer index**, which gives the Series object additional capabilities.

For example, in Pandas, the index doesn't have to be an integer— it can consist of values of any desired type, e.g., we can use strings as an index and the item access works as expected. Here is an **example of a Series based on non-integer index**:

```
import pandas as pd
data = pd.Series([12, 24, 13, 54],
                 index=['a', 'b', 'c', 'd'])

print(data, "\n")
print("Value at index b:", data['b'])
```

2. Series From Dictionaries

We can see that Pandas' Series look much like dictionaries in Python. In fact, we can think of a Pandas Series like a specialization of a Python dictionary. A dictionary is a structure that maps arbitrary keys to a set of arbitrary values, and a Series is a structure that maps typed keys to a set of typed values. This type information makes them more efficient compared to standard dictionaries.

Let's see how we can **create a Series from a dictionary, and then we will**

perform indexing and slicing on it. Say we have a dictionary with keys that are fruits and values that correspond to their amount. We want to use this dictionary to create a Series object and then access values using the names of the fruits:

```
import pandas as pd

fruits_dict = { 'apples': 10,
                'oranges': 8,
                'bananas': 3,
                'strawberries': 20}

fruits = pd.Series(fruits_dict)
print("Value for apples: ", fruits['apples'], "\n")

# Series also supports array-style operations such as slicing:
print(fruits['bananas':'strawberries'])
```