- -> notebooks from this lecture: https://github.com/ine-rmotr-curriculum/data-cleaning-rmotr-freecodecamp
- -> this also works for data frames
- -> how many
- -> to start with an info method
- -> there are four rows
- -> if we need more information about the data
- -> column A in this example has two non-null values
- -> column B has three
- -> one value must be non-null
- -> info gets you close to understanding the amount of data which we have missing
- -> we can also use .sum to find the amount of null values we have in the data frame
- -> drop na <- to drop the rows which have at least one null value
- -> only keeping the column which has no null values
- ***-> to select a subset / threshold <- drop na is a way of doing this***
    - -> drop the rows which have any null values
    - -> or specify a threshold
- ***-> once you have the null values, you can clean / fix them***
    - -> fill the blanks with number 0
    - -> you can drop the rows with na's, or you can replace them with zeroes
    - -> these methods are immutable -> they don't change the original data
    - -> they work by using a for field
    - -> dropping the values
    - -> backwards fields
    - -> this also words for data frames
    - -> for fill axis 1 -> you can also use 0 (which fills the axis vertically)
    - -> row and collumn based
        - -> the values we are replacing can either be done in rows or collumns
- ***-> checking for values***
    - -> is null / the sum method
    - -> we can also use any
    - -> asking if there are any values which are valid, or if all of them are
- ***-> missing values***
    - -> the value is missing (null), there is a hole in it
    - -> we can drop the values / fill the na values
    - -> when there is nothing missing
    - -> d? <- invalid valuss, ones which don't make sense for the context
- ***-> cleaning the data***
    - -> value counts
    - -> this is a summary of all the unique values
    - -> this also gives us a total count for those values
    - -> you can replace those values
    - -> this can work in multiple columns
    - -> we know age and for example we know that 290 is an invalid value for this
        - -> that we will need more programming to clean out these values
        - -> invalid values are ones which don't make sense for the context
- ***-> question***

What will the following code print out?

```
import pandas as pd
import numpy as np
```

```
s = pd.Series([np.nan, 1, 2, np.nan, 3])
s = s.fillna(method='ffill')

print(s)
```

```
0    1.0
1    1.0
2    2.0
3    3.0
4    3.0
dtype: float64
```

```
0    NaN <- This one
1    1.0
2    2.0
3    2.0
4    3.0
dtype: float64
```

```
0    NaN
1    1.0
2    2.0
3    NaN
4    3.0
dtype: float64
```