In [1]:
```
import pandas as pd
import numpy as np
```

```
create a series and print the output
```

In [2]:
```
s=pd.Series([1,2,3,4,5])
s
```

Out[2]:
```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

```
create a dataframe with row and columns
```

In [3]:
```
s=pd.DataFrame(np.random.randn(10,5))
s
```

Out[3]:

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| **0** | 0.656109 | 0.200659 | 0.061140 | 0.291621 | -1.554581 |
| **1** | -1.280845 | 0.777879 | -1.011191 | -0.765897 | -0.651905 |
| **2** | 0.185243 | -0.133339 | 0.747010 | -0.619566 | -1.001455 |
| **3** | -0.949411 | 0.260495 | -0.564892 | 0.456070 | -0.088858 |
| **4** | 0.708772 | 0.914492 | 0.175782 | 1.037809 | 0.166929 |
| **5** | 1.065119 | 0.569842 | -0.908779 | -0.998158 | 0.771977 |
| **6** | 1.113596 | 0.542488 | 0.146780 | 0.015822 | 1.247468 |
| **7** | 0.680481 | -0.365834 | 1.968470 | -1.077360 | -1.021815 |
| **8** | 1.236433 | -0.470777 | -0.550758 | -0.652614 | 0.320818 |
| **9** | -0.108824 | -1.043398 | 0.052701 | -0.461932 | 0.121185 |

```
list the table from last
```

In [4]:
```
s.tail()
```

Out[4]:

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| **5** | 1.065119 | 0.569842 | -0.908779 | -0.998158 | 0.771977 |
| **6** | 1.113596 | 0.542488 | 0.146780 | 0.015822 | 1.247468 |
| **7** | 0.680481 | -0.365834 | 1.968470 | -1.077360 | -1.021815 |
| **8** | 1.236433 | -0.470777 | -0.550758 | -0.652614 | 0.320818 |
| **9** | -0.108824 | -1.043398 | 0.052701 | -0.461932 | 0.121185 |

```
list the table from top
```

In [5]:
```
s.head()
```

Out[5]:

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| **0** | 0.656109 | 0.200659 | 0.061140 | 0.291621 | -1.554581 |
| **1** | -1.280845 | 0.777879 | -1.011191 | -0.765897 | -0.651905 |
| **2** | 0.185243 | -0.133339 | 0.747010 | -0.619566 | -1.001455 |
| **3** | -0.949411 | 0.260495 | -0.564892 | 0.456070 | -0.088858 |
| **4** | 0.708772 | 0.914492 | 0.175782 | 1.037809 | 0.166929 |

```
list all mathematical value
```

In [6]:
```
s.describe()
```

Out[6]:

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| **count** | 10.000000 | 10.000000 | 10.000000 | 10.000000 | 10.000000 |
| **mean** | 0.330667 | 0.125251 | 0.011626 | -0.277421 | -0.169024 |
| **std** | 0.868839 | 0.621251 | 0.879478 | 0.696618 | 0.876136 |
| **min** | -1.280845 | -1.043398 | -1.011191 | -1.077360 | -1.554581 |
| **25%** | -0.035307 | -0.307710 | -0.561359 | -0.737576 | -0.914067 |
| **50%** | 0.668295 | 0.230577 | 0.056921 | -0.540749 | 0.016164 |
| **75%** | 0.976032 | 0.563004 | 0.168532 | 0.222671 | 0.282346 |
| **max** | 1.236433 | 0.914492 | 1.968470 | 1.037809 | 1.247468 |

```
To describe the empty value
```

In [7]:
```
s.isna()
```

Out[7]:

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| **0** | False | False | False | False | False |
| **1** | False | False | False | False | False |
| **2** | False | False | False | False | False |
| **3** | False | False | False | False | False |
| **4** | False | False | False | False | False |
| **5** | False | False | False | False | False |
| **6** | False | False | False | False | False |
| **7** | False | False | False | False | False |
| **8** | False | False | False | False | False |
| **9** | False | False | False | False | False |

```
create a data frame with no empty value
```

In [8]:
```
df1=pd.DataFrame({
    "A":1.0,
    "B":pd.Timestamp("20230721"),
    "C":pd.Series(index=list(range(4)))
})
df1
```

```
<ipython-input-8-1f7f2c8d165e>:4: DeprecationWarning: The default dtype for e
mpty Series will be 'object' instead of 'float64' in a future version. Specif
y a dtype explicitly to silence this warning.
  "C":pd.Series(index=list(range(4)))
```

Out[8]:

| | A | B | C |
|---|---|---|---|
| **0** | 1.0 | 2023-07-21 | NaN |
| **1** | 1.0 | 2023-07-21 | NaN |
| **2** | 1.0 | 2023-07-21 | NaN |
| **3** | 1.0 | 2023-07-21 | NaN |

```
to describe the empty value
```

In [9]:
```
df1.isna()
```

Out[9]:

| | A | B | C |
|---|---|---|---|
| 0 | False | False | True |
| 1 | False | False | True |
| 2 | False | False | True |
| 3 | False | False | True |

```
to fill the empty value by constanst
```

In [10]:
```
df1.fillna(1)
```

Out[10]:

| | A | B | C |
|---|---|---|---|
| 0 | 1.0 | 2023-07-21 | 1.0 |
| 1 | 1.0 | 2023-07-21 | 1.0 |
| 2 | 1.0 | 2023-07-21 | 1.0 |
| 3 | 1.0 | 2023-07-21 | 1.0 |

```
To describe the loc and iloc
```

In [ ]:
```
df1.iloc[3]
```

In [12]:
```
df1.iloc[1:3]
```

Out[12]:

| | A | B | C |
|---|---|---|---|
| 1 | 1.0 | 2023-07-21 | NaN |
| 2 | 1.0 | 2023-07-21 | NaN |

In [16]:
```
df1.dropna()
```

Out[16]:

| A | B | C |
|---|---|---|

In [17]:
```
df1.dropna(axis=1,how="any")
```

Out[17]:

| | A | B |
|---|---|---|
| 0 | 1.0 | 2023-07-21 |
| 1 | 1.0 | 2023-07-21 |
| 2 | 1.0 | 2023-07-21 |
| 3 | 1.0 | 2023-07-21 |

To display columns

In [19]:
```
df1.columns
```

Out[19]:
```
Index(['A', 'B', 'C'], dtype='object')
```

To display row

In [20]:
```
df1.index
```

Out[20]:
```
Int64Index([0, 1, 2, 3], dtype='int64')
```

In [ ]:
```

```