```
abs(-100)         # 100
int(3.1416)       # 3
round(3.1416,2)   # 3.14   #rounds the result to 2 decimal places
pow(2,10)         # 1024   (power function: 2^10 = 1024)
```

```
1024
```

```
abs(-100)         # 100
```

```
100
```

```
int(3.1416)       # 3
```

```
3
```

```
round(3.1416,2)   # 3.14
```

```
3.14
```

```
pow(2,10)         # 1024
```

```
1024
```

```
x = [7, 3, 3, 5, 3, 4, 6, 0, 5, 8, 3, 3, 5, 3, 4, 3, 3, 5, 3, 4]
len(x)   # 20
```

```
20
```

```
import math
math.sqrt(829921)   # 911.0
math.ceil(73.27)    # 74
math.floor(73.27)   # 73
math.pow(2,10)      # 1024
```

```
1024.0
```

```
math.sqrt(829921)   # 911.0
```

```
911.0
```

Start coding or generate with AI.

```
math.ceil(73.27)    # 74
```

```
74
```

Start coding or generate with AI.

```
math.floor(73.27)   # 73
```

```
73
```

```
Start coding or generate with AI.
```

```
math.pow(2,10)      # 1024
```

```
1024.0
```

```
import datetime
now = datetime.datetime.now()
print ("Current date and time : ")
print (now.strftime("%Y-%m-%d %H:%M:%S"))
#>> Current date and time :
#>> 2021-11-16 01:02:32
#
```

```
Current date and time :
2024-03-25 05:10:46
```

```
import time
s = time.time()
print(s)
#>> 1637004821.3311868
```

```
1711343454.8496904
```

```
# Import seaborn library.
# refer: https://seaborn.pydata.org/
import seaborn as sb
# Read the iris data file using the load_dataset method of the seaborn library. Save it in the pandas DataFr
pdf = sb.load_dataset('iris')

# check the data type of ‘pdf’.
type(pdf)
```

```
pandas.core.frame.DataFrame
def __init__(data=None, index: Axes | None=None, columns: Axes | None=None,
dtype: Dtype | None=None, copy: bool | None=None) -> None
```

```
>>> Point = make_dataclass("Point", [("x", int), ("y", int)])
>>> pd.DataFrame([Point(0, 0), Point(0, 3), Point(2, 3)])
   x  y
0  0  0
1  0  3
```

```
# Descriptive statistics for numeric data -
# central tendencies, dispersion etc, excluding 'NaN' (non-null) values
pdf.describe()
```

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
# print the first two rows
pdf.head(2) #print the first 2 rows of the DataFrame
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa

Next steps: Generate code with pdf View recommended plots

```
# print the last two rows
pdf.tail(2) #print the last  2 rows of the DataFrame
```

	sepal_length	sepal_width	petal_length	petal_width	species
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

```
pdf.species # access the column species; same as pdf['species']
pdf['species'] # access the column species; same as pdf.species
```

```
0         setosa
1         setosa
2         setosa
3         setosa
4         setosa
         ...
145    virginica
146    virginica
147    virginica
148    virginica
149    virginica
Name: species, Length: 150, dtype: object
```

```
# Print a summary of the DataFrame - columns, data types, non-null counts
pdf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
```

```
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

```
# filter-1: bool
pdf.species == 'setosa'
```

```
0       True
1       True
2       True
3       True
4       True
       ...
145    False
146    False
147    False
148    False
149    False
Name: species, Length: 150, dtype: bool
```

```
# filter-2: selection of a set of rows based on a logical condition
pdf[pdf.species == 'setosa']
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa
10	5.4	3.7	1.5	0.2	setosa
11	4.8	3.4	1.6	0.2	setosa
12	4.8	3.0	1.4	0.1	setosa
13	4.3	3.0	1.1	0.1	setosa
14	5.8	4.0	1.2	0.2	setosa
15	5.7	4.4	1.5	0.4	setosa
16	5.4	3.9	1.3	0.4	setosa
17	5.1	3.5	1.4	0.3	setosa
18	5.7	3.8	1.7	0.3	setosa
19	5.1	3.8	1.5	0.3	setosa
20	5.4	3.4	1.7	0.2	setosa
21	5.1	3.7	1.5	0.4	setosa
22	4.6	3.6	1.0	0.2	setosa
23	5.1	3.3	1.7	0.5	setosa
24	4.8	3.4	1.9	0.2	setosa
25	5.0	3.0	1.6	0.2	setosa
26	5.0	3.4	1.6	0.4	setosa
27	5.2	3.5	1.5	0.2	setosa
28	5.2	3.4	1.4	0.2	setosa
29	4.7	3.2	1.6	0.2	setosa
30	4.8	3.1	1.6	0.2	setosa

```
# filter-4: selection of a set of rows based on a multiple conditions
# get all rows for species setosa or versicolor
pdf[(pdf['species']=='setosa')   | (pdf['species']=='versicolor')]
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
95	5.7	3.0	4.2	1.2	versicolor
96	5.7	2.9	4.2	1.3	versicolor
97	6.2	2.9	4.3	1.3	versicolor
98	5.1	2.5	3.0	1.1	versicolor
99	5.7	2.8	4.1	1.3	versicolor

100 rows × 5 columns

```
# filter-5: selecting a set of rows based on an inequality condition
pdf[pdf.sepal_length > 7.5]
```

	sepal_length	sepal_width	petal_length	petal_width	species
105	7.6	3.0	6.6	2.1	virginica
117	7.7	3.8	6.7	2.2	virginica
118	7.7	2.6	6.9	2.3	virginica
122	7.7	2.8	6.7	2.0	virginica
131	7.9	3.8	6.4	2.0	virginica
135	7.7	3.0	6.1	2.3	virginica

```
# pandas correlation, rounded to 2 decimals
round(pdf.corr(),2)
```

```
<ipython-input-31-2e3952c51d49>:2: FutureWarning: The default value of numeric_only
  round(pdf.corr(),2)
```

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.00	-0.12	0.87	0.82
sepal_width	-0.12	1.00	-0.43	-0.37
petal_length	0.87	-0.43	1.00	0.96
petal_width	0.82	-0.37	0.96	1.00

```
# sorting can be ascending or descending, with multiple sort keys
pdf.sort_values('species',ascending=False)
pdf.sort_values(['species','sepal_length'],ascending=False)
pdf.sort_values(['species','sepal_length','sepal_width'],ascending=True)
```

	sepal_length	sepal_width	petal_length	petal_width	species
13	4.3	3.0	1.1	0.1	setosa
8	4.4	2.9	1.4	0.2	setosa
38	4.4	3.0	1.3	0.2	setosa
42	4.4	3.2	1.3	0.2	setosa
41	4.5	2.3	1.3	0.3	setosa
...	...	...	...	...	...
118	7.7	2.6	6.9	2.3	virginica
122	7.7	2.8	6.7	2.0	virginica
135	7.7	3.0	6.1	2.3	virginica
117	7.7	3.8	6.7	2.2	virginica
131	7.9	3.8	6.4	2.0	virginica

150 rows × 5 columns

```
# get one random decimal in the range 0 to 1
import numpy as np
np.random.rand()   #  0.44117932776647417

# get one random integer in the range 0 to 100
np.random.randint(100)   # 62

# create a random integer array of 3*5
x = np.random.randint(100, size=(3, 5))
print(x)
```

```
[[79 79 56 99 84]
 [39  0 81 68 80]
 [45 91  5 36 88]]
```

```
import numpy as np
a1 = np.array(['a','b','c',1,2.5,True])
a1.dtype    # dtype('U32')
# items in ‘a1’ are stored as unicode strings with size < = 32 characters
a1.ndim # 1     # 1-dimension
a1.shape    # (6,)      # 6 rows
```

```
(6,)
```

```
import numpy as np
np.ones(5)  #([1., 1., 1., 1., 1.])
```

```
array([1., 1., 1., 1., 1.])
```

```
import numpy as np
a2 = np.array([[1,2,3,4,5], [6,7,8,9,10],['a','b','c','d','e']])

a2[0, 4]   # 5 (element in the cell row:0, column 4)
a2[2, 0]   # a (element in the cell row:3, column 1)
a2[1:3,]   # select all the elements of row-1 and row-2
```

```
array([['6', '7', '8', '9', '10'],
       ['a', 'b', 'c', 'd', 'e']], dtype='<U21')
```

```
a3[2:4]   # ['c' 'd']
```

```
array(['c', 'd'], dtype='<U1')
```

```
print(a3[:5])
print(a3[:99])
```

```
['a' 'b' 'c' 'd' 'e']
['a' 'b' 'c' 'd' 'e']
```

```
import numpy as np
a3 =           np.array(['a','b','c','d','e'])



print(a3[:5])
print(a3[:99])
```

```
['a' 'b' 'c' 'd' 'e']
['a' 'b' 'c' 'd' 'e']
```

```
a4 = np.array([1, 2, 3, 4, 5])
```

```
Start coding or generate with AI.
```

```
a4 = np.array([1, 2, 3, 4, 5])
y = a4.copy()
a4[0] = 666
y
```

```
array([1, 2, 3, 4, 5])
```