PANDAS

```
#import pandas library anf aliasing as pd
import pandas as pd
import numpy as np
data = np.array(['a','b','c','d'])
s = pd.Series(data,index=[100,101,102,103])
print(s)
```

```
100    a
101    b
102    c
103    d
dtype: object
```

```
#series using dictionary
data = {'a':0.,'b':1.,'c':2.}
s = pd.Series(data)
print(s)
```

```
a    0.0
b    1.0
c    2.0
dtype: float64
```

```
data = {'a':0,'b':1.,'c':2.}
s = pd.Series(data,index=['b','c','d','a'])
print(s)
```

```
b    1.0
c    2.0
d    NaN
a    0.0
dtype: float64
```

```
s = pd.Series(5,index=[0,1,2,3])  #the value will be repeated to match the length of index
print(s)
```

```
0    5
1    5
2    5
3    5
dtype: int64
```

```
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])

#retrieve the first element
print(s[0])
```

```
1
```

```
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
print(s)
#retrieve the first three element
print(s[:3])
```

```
a    1
b    2
c    3
d    4
e    5
dtype: int64
a    1
b    2
c    3
dtype: int64
```

```
#Retrieve the single element using index label value
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
#retrieve the single element
print(s['a'])
```

```
1
```

create an integer list , convert it into series and assign string labels to the data elements and assign float data type to it

```
lst = [1,2,3,4,5]
sr = pd.Series(lst, ['a','b','c','d','e'],float) #syntax : pd.Series(list,index,dtype)
print(sr)
```

```
a    1.0
b    2.0
c    3.0
d    4.0
e    5.0
dtype: float64
```

Creating a Data Frame

```
#Creating an empty data frame
import pandas as pd
df = pd.DataFrame()
print(df)
```

```
Empty DataFrame
Columns: []
Index: []
```

Create a Dataframe from lists

```
data = [1,2,3,4,5]
df = pd.DataFrame(data)
print(df)
```

```
   0
0  1
1  2
2  3
3  4
4  5
```

Create a Dataframe from Array

```
data = [["Alex",10],['Bob',12],['Clarke',13]]
df = pd.DataFrame(data,columns = ['Name','Age'])
df
```

	Name	Age
0	Alex	10
1	Bob	12
2	Clarke	13

```
data = [["Alex",10],['Bob',12],['Clarke',13]]
df = pd.DataFrame(data,columns = ['Name','Age'],dtype = float)
df
```

```
<ipython-input-22-ad299877f2f8>:2: FutureWarning: Could not cast to float64, falling back to object. This behavior is deprecated. Ir
  df = pd.DataFrame(data,columns = ['Name','Age'],dtype = float)
```

	Name	Age
0	Alex	10.0
1	Bob	12.0
2	Clarke	13.0

Create a Dataframe from Dictionary

```
import pandas as pd
data = [{'a':1,'b':2},{'a':5, "b":1,"c" : 20}]
df = pd.DataFrame(data)
print(df)
```

```
   a  b     c
0  1  2   NaN
1  5  1  20.0
```

```
#with indexing
data = [{'a':1,'b':2},{'a':5, "b":1,"c" : 20}]
df = pd.DataFrame(data,index=["First","Second"])
print(df)
```

```
        a  b     c
First   1  2   NaN
Second  5  1  20.0
```

```
data = [{'a':1,'b':2},{'a':5, "b":1,"c" : 20}]
#with two column indices, values same as dictionary keys
df1 = pd.DataFrame(data,index =["first","second"],columns=['a','b'])
#with two column indices, with one index with other name
df2 = pd.DataFrame(data,index =["first","second"],columns=['a','b1'])
print(df1)
print(df2)
```

```
        a  b
first   1  2
second  5  1
        a  b1
first   1 NaN
second  5 NaN
```

creating the dataframe from the dictionary with series

```
d = {'one':pd.Series([1,2,3],index=['a','b','c']),
     'two':pd.Series([1,2,3,4],index=['a','b','c','d'])}
df = pd.DataFrame(d)
print(df)
```

```
   one  two
a  1.0    1
b  2.0    2
c  3.0    3
d  NaN    4
```

```
d = {'one':pd.Series([1,2,3],index=['a','b','c']),
     'two':pd.Series([1,2,3,4],index=['a','b','c','d'])}
df = pd.DataFrame(d)
print(df['one'])
```

```
a    1.0
b    2.0
c    3.0
d    NaN
Name: one, dtype: float64
```

Column Addition

```
d = {'one':pd.Series([1,2,3],index=['a','b','c']),
     'two':pd.Series([1,2,3,4],index=['a','b','c','d'])}
df = pd.DataFrame(d)
#Adding a new column to an existing DAtaFrame object with column label by passing as Series
print("Adding a new coulmn by passing as Series: ")
df["three"]=pd.Series([10,20,30],index =['a','b','c'])
print(df)
print("Adding a new column using the existing columns in DataFrame: ")
df["four"]=df['one']+df['three']
print(df)
```

```
Adding a new coulmn by passing as Series:
   one  two  three
a  1.0    1   10.0
b  2.0    2   20.0
c  3.0    3   30.0
d  NaN    4    NaN
Adding a new column using the existing columns in DataFrame:
   one  two  three  four
a  1.0    1   10.0  11.0
b  2.0    2   20.0  22.0
c  3.0    3   30.0  33.0
d  NaN    4    NaN   NaN
```

```
#Using the previous DAtFrame,we will delete a column
#using del function
import pandas as pd
d = {'one':pd.Series([1,2,3],index=['a','b','c']),
     'two':pd.Series([1,2,3,4],index=['a','b','c','d']),
     'three':pd.Series([10,20,30],index=['a','b','c'])}
df = pd.DataFrame(d)
print("Our dataframe is:")
print(df)

#using del function
print("Deleting the first column using DEL function:")
del (df['one'])
print(df)
#using pop function
df.pop('two')
print(df)
```

```
Our dataframe is:
   one  two  three
a  1.0    1   10.0
b  2.0    2   20.0
c  3.0    3   30.0
d  NaN    4    NaN
Deleting the first column using DEL function:
   two  three
a    1   10.0
b    2   20.0
c    3   30.0
d    4    NaN
   three
a   10.0
b   20.0
c   30.0
d    NaN
```

```
d = {'one':pd.Series([1,2,3],index=['a','b','c']),
     'two':pd.Series([1,2,3,4],index=['a','b','c','d'])}
df = pd.DataFrame(d)
print(df.loc['b'])
```

```
one    2.0
two    2.0
Name: b, dtype: float64
```

```
d = {'one':pd.Series([1,2,3],index=['a','b','c']),
     'two':pd.Series([1,2,3,4],index=['a','b','c','d'])}
df = pd.DataFrame(d)
print(df[2:4])
```

```
   one  two
c  3.0    3
d  NaN    4
```

Addition of rows

```
df = pd.DataFrame([[1,2],[3,4]],columns = ['a','b'])
df2 = pd.DataFrame([[5,6],[7,8]],columns = ['a','b'])
df = df.append(df2)
print(df)
```

```
   a  b
0  1  2
1  3  4
0  5  6
1  7  8
<ipython-input-25-a2d676ca3e2f>:3: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future
  df = df.append(df2)
```

```
#Drop rows with label 0
df = df.drop(0)
print(df)
```

```
   a  b
1  3  4
1  7  8
```

Load the Dataset

```
import pandas as pd
df = pd.read_csv("/content/Heart disease dataset_SET B.csv")
```

```
df.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
df.tail()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

```
df.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.00000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.39934
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.61622
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.00000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.00000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.00000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.00000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.00000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

```
df.shape
```

```
(303, 14)
```

```
df.columns
```

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

```
df.tail(5)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

Start coding or generate with AI.