In [17]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Introduction Pandas

import pandas as pd
```

In [18]:
```
# creating pandas Series

numbers = range(1,100,5)
pd.Series(numbers)
```

Out[18]:
```
0      1
1      6
2     11
3     16
4     21
5     26
6     31
7     36
8     41
9     46
10    51
11    56
12    61
13    66
14    71
15    76
16    81
17    86
18    91
19    96
dtype: int64
```

In [19]:
```
# create a oject series
string = "Hi", "How", "are", "you", "?"
pd.Series(string)
```

Out[19]:
```
0     Hi
1    How
2    are
3    you
4      ?
dtype: object
```

In [20]:
```
# creat a Series with an arbitrary list

s = pd.Series([345, 'London', 34.5, -34.45, 'Happy Birthay'])
s
```

Out[20]:
```
0             345
1          London
2            34.5
3          -34.45
4    Happy Birthay
dtype: object
```

In [21]:
```
# to set index values for a series

marks = [60, 89, 74, 86]

subject = ["Maths", "Science", "English", "Social Science"]
pd.Series(marks, index = subject)
```

Out[21]:
```
Maths             60
Science           89
English           74
Social Science    86
dtype: int64
```

In [22]:
```
# To create a series from a dictionary

data = {'Maths': 60, 'Science': 80, 'English': 76, 'Social Science': 86}
pd.Series(data)
```

Out[22]:
```
Maths             60
Science           80
English           76
Social Science    86
dtype: int64
```

In [23]:
```
# A series with missing values

subject = ["Maths", "Science", "Art and Craft", "Social Science"]
marksSeries = pd.Series(data, index = subject)
(print(marksSeries))
```

```
Maths             60.0
Science           80.0
Art and Craft      NaN
Social Science    86.0
dtype: float64
```

In [25]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Introduction Pandas

import pandas as pd
index = ['Apple', 'Banana', 'Orange']
quantity = [34, 20, 30, 40]
pd.Series(data=quantity, index=subject)
```

Out[25]:
```
Maths             34
Science           20
Art and Craft     30
Social Science    40
dtype: int64
```

In [14]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Series

import pandas as pd
dict = {'A':30, 'B':40, 'C':50}
```

```
index = ['A', 'B', 'D']
pd.Series(data=dict, index=index)
```

Out[14]:
```
A    30.0
B    40.0
D     NaN
dtype: float64
```

In [26]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Series

import pandas as pd
s1 = pd.Series([1, 2, 5, 6.5])
s2 = pd.Series(['first', 35, 'college', 62.5])
s1
```

Out[26]:
```
0    1.0
1    2.0
2    5.0
3    6.5
dtype: float64
```

In [27]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Series
s2
```

Out[27]:
```
0      first
1         35
2    college
3       62.5
dtype: object
```

In [28]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series

# to check for null values using isnull
marksSeries.isnull()
```

Out[28]:
```
Maths             False
Science           False
Art and Craft      True
Social Science    False
dtype: bool
```

In [29]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series

# to check for null values using notnull
marksSeries.notnull()
```

Out[29]:
```
Maths              True
Science            True
Art and Craft     False
Social Science     True
dtype: bool
```

In [30]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series

# to know the subjeccts in which sccore is more than 75
marksSeries[marksSeries >75]
```

Out[30]:
```
Science           80.0
Social Science    86.0
dtype: float64
```

In [31]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series

# to assign 68 marks to 'Art and Craft'

marksSeries["Art and Craft"] = 68
marksSeries
```

Out[31]:
```
Maths             60.0
Science           80.0
Art and Craft     68.0
Social Science    86.0
dtype: float64
```

In [32]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series

# to check whether Maths marks are 73
marksSeries.Maths ==73
```

Out[32]:
```
False
```

In [22]:
```
# or you may use
marksSeries["Maths"]== 73
```

Out[22]:
```
False
```

In [33]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series - Sorting a numeric series

# create a padas series
values = pd.Series([23, 45, 41, 23, 34, 55, 34, 20])
values
```

Out[33]:
```
0    23
1    45
2    41
3    23
4    34
5    55
6    34
7    20
dtype: int64
```

In [34]:
```
# ascending order
values.sort_values(ascending = True)
```

Out[34]:
```
7    20
0    23
3    23
4    34
6    34
2    41
1    45
5    55
dtype: int64
```

In [35]:
```
# descending order
values.sort_values(ascending = False)
```

Out[35]:
```
5    55
1    45
2    41
4    34
6    34
0    23
3    23
7    20
dtype: int64
```

In [36]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series - Sorting a categories series

# create a padas series
stringValues = pd.Series(["a", "j", "d", "f", "t", "a"])
stringValues
```

Out[36]:
```
0    a
1    j
2    d
3    f
4    t
5    a
dtype: object
```

In [37]:
```
# ascending order
stringValues.sort_values(ascending = True)
```

Out[37]:
```
0    a
5    a
2    d
3    f
1    j
4    t
dtype: object
```

In [38]:
```
# descending order
stringValues.sort_values(ascending = False)
```

Out[38]:
```
4    t
1    j
3    f
2    d
0    a
5    a
dtype: object
```

In [39]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Manipulating Series – Rank Series

marksSeries.rank(ascending = True, pct=False)
```

Out[39]:
```
Maths             1.0
Science           3.0
Art and Craft     2.0
Social Science    4.0
dtype: float64
```

In [45]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Manipulating Series

import pandas as pd
data = [0.85, 0.8, 0.98, 0.74, 0.4, 0.55, 0.94, 0.42, 0.43, 0.92]
ser = pd.Series(data=data)
```

In [46]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Manipulating Series

ser.sort_values(ascending=False)
```

Out[46]:
```
2    0.98
6    0.94
9    0.92
0    0.85
1    0.80
3    0.74
5    0.55
8    0.43
7    0.42
4    0.40
dtype: float64
```

In [49]:
```
data = range(10)
new_ser = pd.Series(data=data)
new_ser[new_ser==5]
```

Out[49]:
```
5    5
dtype: int64
```

In [50]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Introduction to Dataframes and Creating DataFrame

import pandas as pd
a1 = ['Hogwarts', 'Durmstrang', 'Beauxbatons']
a2 = ['Hogwarts', 'Durmstrang', 'Beauxbatons']
```

```
a3 = ['Hogwarts', 'Durmstrang', 'Beauxbatons']
school = [a1, a2, a3]
inst = ['School_1', 'School_2', 'School_3']
Muggle_data = pd.DataFrame(data=school, columns=inst)
Muggle_data
```

Out[50]:

	School_1	School_2	School_3
0	Hogwarts	Durmstrang	Beauxbatons
1	Hogwarts	Durmstrang	Beauxbatons
2	Hogwarts	Durmstrang	Beauxbatons

In [51]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Introduction to Dataframes and Creating DataFrame

import pandas as pd

data = {'A':[1,2,3,4,5], 'B':[1,0,1,1,0]}
df = pd.DataFrame(data=data)
#df.C = df.A + df.B
```

In [46]:
```
# Aut@r: Susana Edith Barrientos Galicia
# concatenate Pandas Series

import pandas as pd

seriesA = pd.Series([101, 102, 103, 104, 105, 106])
seriesB= pd.Series([107, 108, 109, 110, 111, 112])

# concatenate the pandas series
pd.concat([seriesA, seriesB])
```

Out[46]:
```
0    101
1    102
2    103
3    104
4    105
5    106
0    107
1    108
2    109
3    110
4    111
5    112
dtype: int64
```

In [48]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Add a Hierarchical Index on Pandas Series

pd.concat([seriesA, seriesB], keys = ['a', 'b'])
```

Out[48]:
```
a  0    101
   1    102
   2    103
   3    104
   4    105
   5    106
b  0    107
   1    108
   2    109
   3    110
   4    111
   5    112
dtype: int64
```

In [53]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Label the Index

pd.concat([seriesA, seriesB], keys = ['a', 'b'], names=['Series', 'Row ID'])
```

Out[53]:
```
Series  Row ID
a       0         101
        1         102
        2         103
        3         104
        4         105
        5         106
b       0         107
        1         108
        2         109
        3         110
        4         111
        5         112
dtype: int64
```

In [86]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Label the Index

df1 = pd.DataFrame({
    'Name': ['ebay', 'edwin', 'suba', 'praha', 'jon'],
    'Compony': ['Apple', 'Walmart', 'Intel', 'cummins', 'Ford'],
    'Salary' : [67000, 90000, 87000, 69000, 78000]},
    index=[101, 102, 103, 104, 105])

print("The first dataframe is : \n",df1, "\n\n")

df2 = pd.DataFrame({
    'Name': ['Billy', 'Brian', 'Bran', 'Bryce', 'Betty'],
    'Compony': ['Apple', 'Walmart', 'Intel', 'cummins', 'Ford'],
    'Salary' : [89000, 80000, 79000, 97000, 88000]},
    index=[101, 102, 103, 104, 105])

print("The second dataframe is: \n", df2)
```

```
The first dataframe is :
      Name  Compony  Salary
101   ebay    Apple   67000
102  edwin  Walmart   90000
103   suba    Intel   87000
104  praha  cummins   69000
105    jon     Ford   78000


The second dataframe is:
      Name  Compony  Salary
101  Billy    Apple   89000
102  Brian  Walmart   80000
103   Bran    Intel   79000
104  Bryce  cummins   97000
105  Betty     Ford   88000
```

In [87]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Concatenating Pandas Dataframes using .concat()

print(pd.concat([df1,df2]))
```

```
     Name  Compony  Salary
101   ebay    Apple   67000
102  edwin  Walmart   90000
103   suba    Intel   87000
104  praha  cummins   69000
105    jon     Ford   78000
101  Billy    Apple   89000
102  Brian  Walmart   80000
103   Bran    Intel   79000
104  Bryce  cummins   97000
105  Betty     Ford   88000
```

In [64]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Concatenating Pandas Dataframes Horizontally

pd.concat([df1, df2], axis=1)
```

Out[64]:

	Name	Compony	Salary	Name	Compony	Salary
101	ebay	Apple	67000	Billy	Apple	89000
102	edwin	Walmart	90000	Brian	Walmart	80000
103	suba	Intel	87000	Bran	Intel	79000
104	praha	cummins	69000	Bryce	cummins	97000
105	jon	Ford	78000	Betty	Ford	88000

In [88]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Concatenating data frames ignoring index values

pd.concat([df1, df2], ignore_index=True)
```

Out[88]:

	Name	Company	Salary
0	ebay	Apple	67000
1	edwin	Walmart	90000
2	suba	Intel	87000
3	praha	cummins	69000
4	jon	Ford	78000
5	Billy	Apple	89000
6	Brian	Walmart	80000
7	Bran	Intel	79000
8	Bryce	cummins	97000
9	Betty	Ford	88000

In [69]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Concatenating data frames ignoring index values

import numpy as np
import pandas as pd
ser1 = pd.Series(list('abcd'))
ser2 = pd.Series(np.arange(4))
```

In [70]:
```
df = pd.concat([ser1, ser2], axis=1)
```

In [71]:
```
df
```

Out[71]:

	0	1
0	a	0
1	b	1
2	c	2
3	d	3

In [71]:
```
df = pd.concat([ser1, ser2], axis=0)
df
```

Out[71]:
```
0    a
1    b
2    c
3    d
0    0
1    1
2    2
3    3
dtype: object
```

In [72]:
```
# Aut@r: Susana Edith Barrientos Galicia
# Concatenating data frames ignoring index values

import pandas as pd
data1={'Physics': [77, 75, 100, 10, 59], 'Chemistry': [85, 70, 99, 30, 80]}
df1=pd.DataFrame(data=data1)
data2={'Student_ID': [0, 1, 2, 3, 4], 'Maths': [80, 90, 88, 25, 90]}
df2=pd.DataFrame(data=data2)
df3=pd.concat([df1, df2], join='inner', axis=0, ignore_index=True)
```

In [75]:
```
df3
```

Out[75]:

0
1
2
3
4
5
6
7
8
9

In [73]:
```
# Aut@r: Susana Edith Barrientos Galicia
# data frames ignoring index values

import pandas as pd
data1={'Student_ID': [3, 4, 6, 8, 10], 'CGPA': [4.5, 3, 4.37, 3.5, 4]}
df1=pd.DataFrame(data=data1)
data2={'Student_ID': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
'Maths': [4,52, 5, 2.5, 3, 3.9, 2.8, 4.75, 3.68, 5, 4.8]}
df2=pd.DataFrame(data=data2)
df3=pd.merge(df2, df1, on='Student_ID', how='left')
```

In [77]:
```
df3
```

Out[77]:

	Student_ID	Maths	CGPA
0	0	4.00	NaN
1	1	52.00	NaN
2	2	5.00	NaN
3	3	2.50	4.50
4	4	3.00	3.00
5	5	3.90	NaN
6	6	2.80	4.37
7	7	4.75	NaN
8	8	3.68	3.50
9	9	5.00	NaN
10	10	4.80	4.00

In [94]:
```
# Aut@r: Susana Edith Barrientos Galicia
# data frames index values

import pandas as pd
header = pd.MultiIndex.from_product([['Before Course','After Course'],['Mark
d=([[82,95],[78,89]])

my_df = pd.DataFrame(d,
 index=['Alisa','Bobby'],
 columns=header)

print(my_df.stack(level=0).unstack(level=0))
```

```
             Marks
             Alisa Bobby
After Course    95    89
Before Course   82    78
```

```
/var/folders/n2/6lttq_gx5czgqryl7xqhv9800000gp/T/ipykernel_96187/2918254284.
py:12: FutureWarning: The previous implementation of stack is deprecated and
will be removed in a future version of pandas. See the What's New notes for
pandas 2.1.0 for details. Specify future_stack=True to adopt the new impleme
ntation and silence this warning.
  print(my_df.stack(level=0).unstack(level=0))
```

In [93]:
```
# Aut@r: Susana Edith Barrientos Galicia
# data frames index values

import pandas as pd
df = pd.DataFrame({"Gender": ["Male", "Male", "Female", "Female", "Female",
 "Movie_Genre": ["Action", "Comedy", "Drama", "Action", "Comedy", "Drama", "
 "Rating": [1, 5, 3, 2, 3, 4, 4, 5, 4]})
```

In [78]:
```
pd.pivot_table(data=df, index='Gender', values='Rating')
```

Out[78]:

Gender	Rating
Female	3.20
Male	3.75

In [92]:
```
pd.pivot_table(data=df, index='Gender', values='Rating', aggfunc='mean')
```

Out[92]:

Gender	Rating
Female	3.20
Male	3.75

In [83]:
```
pd.pivot_table(data=df, index='Gender', values='Rating')
```

Out[83]:

Gender	Rating
Female	3.20
Male	3.75

In [84]:
```
pd.pivot_table(data=df, index='Gender' , values='Rating', aggfunc='sum')
```

Out[84]:

Gender	Rating
Female	16
Male	15

In [ ]:
```
pd.pivot_table(data=df, index='Gender', 'Movie_Genre' , values='Rating', agg
```

In [86]:
```
pd.pivot_table(data=df, index=['Gender', 'Movie_Genre'] , values='Rating', a
```

Out[86]:

		Rating
Gender	**Movie_Genre**	
Female	**Action**	6
	Comedy	3
	Drama	7
Male	**Action**	5
	Comedy	5
	Drama	5

In [87]:
```
pd.pivot_table(data=df, index=['Gender', 'Movie_Genre'] , values='Rating')
```

Out[87]:

		Rating
Gender	**Movie_Genre**	
Female	**Action**	3.0
	Comedy	3.0
	Drama	3.5
Male	**Action**	2.5
	Comedy	5.0
	Drama	5.0

In [88]:
```
pd.pivot_table(data=df, index='Gender' , values='Rating', aggfunc='sum')
```

Out[88]:

	Rating
Gender	
Female	16
Male	15

In [89]:
```
# correcto

pd.pivot_table(data=df, index=['Gender', 'Movie_Genre'] , values='Rating', a
```

Out[89]:

		Rating
Gender	**Movie_Genre**	
Female	**Action**	6
	Comedy	3
	Drama	7
Male	**Action**	5
	Comedy	5
	Drama	5

In [90]:
```
# Aut@r: Susana Edith Barrientos Galicia
# data frames index values

import pandas as pd

df_employee = [('John', 3400, 'Sydeny'),
 ('Robert', 3000, 'Chicago'),
 ('Aadi', 1600, 'New York'),
 ('Robert', 3000, 'Chicago'),
 ('Robert', 3000, 'Chicago'),
 ('Robert', 3000, 'Texas'),
 ('Aadi', 4000, 'London'),
 ('Sachin', 3000, 'Chicago')]

df_employee = pd.DataFrame(df_employee, columns=['Name', 'Salary', 'City'])

df_employee[df_employee.duplicated('Name')].shape
```

Out[90]: (4, 3)

In [101…
```
# Aut@r: Susana Edith Barrientos Galicia
# Map and Replace

dfOne = pd.DataFrame({
    'Country':['China', 'India', 'Usa', 'Indonesia', 'Brazil'],
    'Population' : [1403500365, 1324171354,322179605,261115456,2076652865]
})
```

In [103…
```
dfOne
```

Out[103…

	Country	Population
0	China	1403500365
1	India	1324171354
2	Usa	322179605
3	Indonesia	261115456
4	Brazil	2076652865

In [106…

```
# create a directory
capital = {
    'Germany':'Berlin',
    'Brazil':'Brasilia',
    'Hungary':'Budapest',
    'China':'Beijing',
    'India':'New Delhi',
    'Norway': 'Oslo',
    'Francia':'Paris',
    'Indonesia':'Jakarta',
    'USA':'Washington',
}

dfOne['Capital']=dfOne['Country'].map(capital)
dfOne
```

Out[106…

	Country	Population	Capital
0	China	1403500365	Beijing
1	India	1324171354	New Delhi
2	Usa	322179605	NaN
3	Indonesia	261115456	Jakarta
4	Brazil	2076652865	Brasilia

In [109…

```
# Aut@r: Susana Edith Barrientos Galicia
# Map and Replace

df7 = pd.DataFrame({
    'col1':[23,10,20],
    'col2':[67,30,56]},
    index=[1,2,3])

print(df7)
```

```
   col1  col2
1    23    67
2    10    30
3    20    56
```

In [112…

```
dict1 = {10:"A", 20:"B"}
```

```
df7['col1'].replace(dict1, inplace=True)
print(df7)
```

```
  col1  col2
1   23    67
2    A    30
3    B    56
```

In [130…
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Map and Replace

import pandas as pd
data = {'Student1': {'name': 'Emma', 'age': '27', 'sex': 'Female'},
 'Student2': {'name': 'Mike', 'age': '22', 'sex': 'Male'}}
df_students = pd.DataFrame(data=data)
df_students
```

Out[130…

	Student1	Student2
name	Emma	Mike
age	27	22
sex	Female	Male

In [131…
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Map and Replace

df_students['Student2'].replace('Mike', 'John', inplace=True)
df_students
```

Out[131…

	Student1	Student2
name	Emma	John
age	27	22
sex	Female	Male

In [132…
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Map and Replace

df_students.replace(['Mike', 'John'], inplace=True)
df_students
```

Out[132…

	Student1	Student2
name	Emma	John
age	27	22
sex	Female	Male

In [134…
```
# Aut@r: Susana Edith Barrientos Galicia
# Test Map and Replace
```

```
df_students.replace(['Mike', 'John'], inplace=True)
df_students
```

Out[134…

	Student1	Student2
name	Emma	John
age	27	22
sex	Female	Male

In [137…

```
# Aut@r: Susana Edith Barrientos Galicia
# Test Map and Replace

df_students['Student2']=df_students['Student2'].map({'Mike':'John'})
df_students
```

Out[137…

	Student1	Student2
name	Emma	NaN
age	27	NaN
sex	Female	NaN

In [148…

```
# Aut@r: Susana Edith Barrientos Galicia
# Groupoy in Pandas


#create d data frame
import pandas as pd
dataFrame=pd.DataFrame({
    'Product_ID':[101,102,103,104,105,106],
    'Food_Product':['Cakes','Biscuis','Fruit','Beverages','Cakes','Beverages
    'Brand':['Baskin Robins','Blue Riband','Peach','Horlicks','Mars Muffin',
    'Sales':[5000, 8000, 7600, 5500, 6500, 9000],
    'Profit':[55000, 67000, 89000, 78000, 55000,90000]})
print(dataFrame)
```

```
   Product_ID Food_Product          Brand  Sales  Profit
0         101        Cakes  Baskin Robins   5000   55000
1         102      Biscuis    Blue Riband   8000   67000
2         103        Fruit          Peach   7600   89000
3         104    Beverages       Horlicks   5500   78000
4         105        Cakes    Mars Muffin   6500   55000
5         106    Beverages        Miranda   9000   90000
```

In [149…

```
dataFrame
```

Out[149…

	Product_ID	Food_Product	Brand	Sales	Profit
0	101	Cakes	Baskin Robins	5000	55000
1	102	Biscuis	Blue Riband	8000	67000
2	103	Fruit	Peach	7600	89000
3	104	Beverages	Horlicks	5500	78000
4	105	Cakes	Mars Muffin	6500	55000
5	106	Beverages	Miranda	9000	90000

In [153…
```
# Number of Unique Column Values Per Group
dataFrame.groupby("Food_Product")["Sales"].nunique().to_frame()
```

Out[153…

Food_Product	Sales
Beverages	2
Biscuis	1
Cakes	2
Fruit	1

In [154…
```
# Sort Groupby Results
dataFrame.groupby("Food_Product")["Sales"].sum().to_frame().reset_index()
```

Out[154…

	Food_Product	Sales
0	Beverages	14500
1	Biscuis	8000
2	Cakes	11500
3	Fruit	7600

In [155…
```
# Sort Groupby Results
dataFrame.groupby("Food_Product")["Sales"].sum().to_frame().reset_index().sc
```

Out[155…

	Food_Product	Sales
3	Fruit	7600
1	Biscuis	8000
2	Cakes	11500
0	Beverages	14500

In [156…
```
# Sort Groupby Results
```

```
dataFrame.groupby("Food_Product").agg({'Sales':['min', 'max', 'mean']})
```

Out[156…

	Sales		
	min	max	mean
Food_Product			
Beverages	5500	9000	7250.0
Biscuis	8000	8000	8000.0
Cakes	5000	6500	5750.0
Fruit	7600	7600	7600.0

In [ ]: